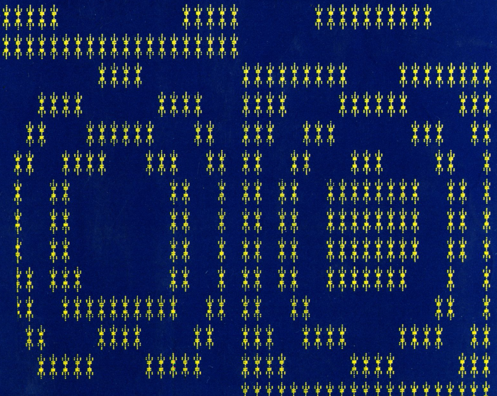
パソコン ライブラリ＝1

# 基本BASICとその応用

戸川隼人＝著

サイエンス社



〈パソコン ライブラリ〉

パソコンライブラリは「論理的で明快な説明・適切な例題・参照に便利なページ単位構成」を特色とし，著者の多年にわたるコンピュータ教育の経験をもとに，独習書としても，教材としても最適です．また「オリジナルなプログラム例が豊富」・「実務にすぐ役立つプログラム技法に詳しい」・「グラフィックスに詳しい」・「文字列処理の説明がわかり易い」・「エラー・メッセージの詳しい解説がある」というので，ベテラン・ユーザーの間でも「資料価値がある」と好評のシリーズです．

| | |
|---|---|
| ①基本BASICとその応用 | 定価1906円 |
| ②各種パソコン用 拡張BASICとその応用 | 定価1900円 |
| ③マイクロソフトBASIC詳説 | 定価3605円 |
| ⑤PC-8001 BASIC | 定価1648円 |
| ⑥PC-8801 BASIC | 定価2000円 |
| ⑦16ビットパソコン PC-9801 BASIC | 定価2500円 |
| ⑧Level 3／Mark II BASIC | 定価1751円 |
| ⑨FM-7 BASIC | 定価2060円 |
| ⑩FM-8 BASIC | 定価1751円 |
| ⑪FM-11 BASIC | 定価2472円 |
| ⑫パソコンテレビ X1 BASIC | 定価2472円 |
| ⑬ポケット・コンピューター PC-1500 BASIC | 定価1339円 |
| ⑭MZ-700 HuBASIC | 定価1854円 |
| ⑮MZ-80B, 2000 HuBASIC | 定価2575円 |
| ⑯PASOPIA 7 BASIC | 定価1854円 |
| ⑰おしゃべりパソコン PC-6001mk II BASIC | 定価2060円 |
| ⑱if-800 モデル50 BASIC | 定価2369円 |
| ⑲MZ-2200 HuBASIC | 定価2266円 |
| ⑳PC-9801F 日本語BASIC | 定価2680円 |
| ㉑PC-8801mk II ディスクBASIC | 定価2678円 |
| ㉒IBM-5550 BASIC | 定価2680円 |
| ㉓PC-UX入門 | 定価1648円 |
| ㉔X1 turbo入門 | 定価1350円 |
| ㉕X1 turbo BASIC基礎編 | 定価1500円 |
| ㉖X1 turbo グラフィックス | 定価1800円 |
| ㉗PC-8801mk II SR BASIC | 定価2678円 |
| ㉘PC-9800シリーズVM/VX/XL BASIC | 定価2890円 |

※表示価格は全て税込みです．

| 機　能 | 書きかた（灰色の所は書かなくてもよい） |
|---|---|
| 注　釈 | REM 文字列<br>(例) REM EXAMPLE |
| 配列の宣言 | DIM 配列名（寸法），配列名（寸法），…<br>(例) DIM A(20,20),X(15)<br>(注) 寸法10以下ならば宣言不要 |
| 配列の下限の指定 | ŌPTIŌN 下限の値<br>(例) ŌPTIŌN 1<br>(注) 指定しなければ下限は0 |
| 代　入 | LET 代入先の変数名＝式<br>(例) A=B+C |
| 飛び越し | GŌTŌ 行先の行番号<br>(例) GŌTŌ 246 |
| 反　復 | FŌR 制御変数名＝始値 TŌ 終値 STEP 増分<br>(例) FŌR I=1 TŌ 10 |
| 反復の範囲の指定 | NEXT 制御変数名<br>(例) NEXT I |
| 判断，分岐 | IF 等式または不等式 THEN 行先の行番号<br>(例) IF A=B THEN 254 |
| キー番号による分岐 | ŌN キー番号 GŌTŌ 行先の行番号の列<br>(例) ŌN K GŌTŌ 100,300,800 |
| サブルーチンに飛ぶ | GŌSUB 入口の文の行番号<br>(例) GŌSUB 7000 |
| サブルーチンの出口 | RETURN |
| 関数の定義 | DEF 関数名（引数）＝式<br>(例) FNC(X)=2*X+1 |
| 乱数列の初期化 | RANDŌMIZE |

| 機　能 | 書きかた（灰色の所は書かなくてもよい） |
|---|---|
| 端末機からの入力 | INPUT 変数名をコンマで区切って書く<br>(例) INPUT X,Y,A(1) |
| プログラムに付けたデータの入力 | READ 変数名をコンマで区切って書く<br>(例) READ N,H(K),I¥ |
| データの定義 | DATA 値をコンマで区切って書く<br>(例) DATA 8,1.5,"ABC" |
| 最初のデータに戻る | RESTŌRE |
| 出　力 | PRINT 式をコンマまたはセミコロンで区切って書く<br>(例) PRINT A,X+1,"MŌZI",I¥<br>(注) コンマで区切ると上下のそろった出力形式，セミコロンで区切るとコンパクトな出力形式になる．ただし両者を混用した場合はその限りでない． |
| 指定印字位置までのスキップ | TAB（印字位置）をPRINT文の列中に書く<br>(例) PRINT TAB(12);A |
| 一時停止 | STŌP<br>(注) CŌNTINUEコマンドで続きの実行再開可能 |
| 終　了 | END |

| | | | |
|---|---|---|---|
| 使用できる文字 | 英字，数字および下記の記号<br>＋－＊／＾＝＜＞（）" '：；．，＿！＃¥％＆？ | | |
| 算術演算子 | ＋－＊／＾ | | |
| 比較演算子 | ＜　＜＝　＝　＞＝　＞　＜＞ | | |
| 組み込み関数 | SIN　$\sin x$ | SQR | 平方根 |
| | CŌS　$\cos x$ | ABS | 絶対値 |
| | TAN　$\tan x$ | SGN | 符号 |
| | ATN　$\arctan x$ | INT | 整数化 |
| | EXP　$e^x$ | RND | 乱数 |
| | LŌG　$\log_e x$ | | |

パソコン ライブラリ＝1

# 基本BASICとその応用

戸川隼人＝著

サイエンス社

```
 10 DEF FNC(X)=SIN(X)*CŌS(X)
 20 PRINT "A,B,C,D,F0,F1="
 30 INPUT  A,B,C,D,F0,F1
 40 LET H=(B-A)/40
 50 LET K=(D-C)/40
 60 LET W=(F1-F0)/10
 70 FŌR J=0 TŌ 40
 80    LET Y=C+(40-J)*K
 90    FŌR I=0 TŌ 40
100       LET X=A+I*H
110       LET Z=FNC(X,Y)
120       LET T=(Z-F0)/W
130       FŌR M=0 TŌ 10 STEP 2
140          IF T<M   THEN 170
150          IF T<M+1 THEN 190
160       NEXT M
170       PRINT "␣";
180       GŌTŌ 200
190       PRINT "*";
200    NEXT I
210    PRINT
220 NEXT J
230 END
```

# ま え が き

この本は，

コンピューターの基礎知識

BASIC の文法

基本的なプログラミング技法

数値計算法の初歩

を例題中心に説明し，たくさんの応用問題とその解答を付けたものである。もともと，大学等におけるコンピューター教育のテキスト，参考書，演習指導書として作成したものであるが，ていねいな説明が付いているので独習書としてもご利用いただけると思う。

この本のねらいは，BASIC という特定の言語に習熟させることよりも，むしろ「BASIC を通してコンピューターを理解させる」ことにある。そのため，文法よりもコンピューターの活用法に重点を置いている。BASIC という言語は簡単で覚え易く，このような目的にはまさに好適である。演習問題も，無味乾燥な文法は最小限にとどめ，応用の基本的なパターンを数多く収録したつもりである。

本書で用いる BASIC の文法の範囲は，国際標準規格でいうところの基本 BASIC (Minimal BASIC) に限定した。高級マイコンの BASIC の仕様は，これよりもはるかにデラックスであるから，本書の内容は，高級マイコンの使用者にとってはおそらく不満が多いことと思う。しかし，教育機関においては基本 BASIC しか使えない機種がまだ数多く使用されているので，そういう機種でも実習できるように配慮したわけである。マイコンで使われている拡張 BASIC に関しては，いずれそのうちに，専用のテキストを作りたいと考えている。

## 謝　辞

株式会社日立製作所のご好意により，装置の内部や最新の機器の写真を多数使わせていただきました。お骨折りいただいたコンピュータ事業部教育センター部の水原邦保氏はじめ関係の方々に御礼申し上げます。また，出版に際し一方ならぬお世話になったサイエンス社の森平勇三氏，橋元淳一郎氏，富澤　昇氏，古瀬美津雄氏に御礼申し上げます。

著　者

## 演習問題解答のこと

独習される方々のために，演習問題解答を付けました。これは，あくまでも参考用で，このとおりでなければいけない，というわけではありません。もっとうまいプログラムも作れると思います。ここにあげたプログラムは，全部コンピューターでテストしてあります(主に BASIC Master Level-3 を使用，一部は PC-6001 を使用)。しかし，機種による扱い方の違いがいくらかありますし，テストも完全とはいえないと思います。お気付きの点がございましたら，ご一報いただければ幸いです。

戸川　保子

# 目　　次

# 1 コンピューターの概要

## 1.1 コンピューターとは

コンピューターは情報を扱う機械である。

たとえば，情報の

　　入力　　収集　　記憶　　伝送　　出力

　　分類　　照合　　検索　　変換　　計算

などを行なうことができる。

コンピューターは情報を符号列（文字列）の形で扱う。そういう扱いかたを**デジタル**（digital）方式という。情報の扱いかたとしては，そのほかに**アナログ**（analogue）という方式がある。これは量的情報を量に比例（ないしは連続的に対応）する物理量で表すやりかた（たとえば普通の寒暖計，レコード，写真など）である。デジタル方式はアナログ方式に比較して，概して機構が複雑になるが，

　　より広範囲の情報を扱うことができる

　　複雑な処理が可能

などの利点がある。

コンピューターのもう一つの特長は，

　　装置は単純かつ汎用的にしてある

　　その動作順序を**プログラム**という形でセットできる

　　そのプログラムをまた情報として処理することが可能

ということである。これを**プログラム内蔵**（stored program）**方式**という。その結果として，装置を複雑にすることなく，きわめて複雑な処理が可能になっており，また汎用機械として多様な目的に使用できるわけである。

## 1.2 原　　理

コンピューターで扱う情報には

- 数値
- 文字列
- 符号パターン（たとえば画像をそのまま符号化したもの）

などがあるが，最終的にはそれらをすべて0と1の列

（例）　1 0 1 1 0 1 0 …

によって表現し処理する。それは

たった2種類の符号「0」「1」を処理できればよい

したがって機構が簡単になる

組合せによって各種の数値や文字やパターンを表現できる

という利点があるからである。0と1を表現するには，要するに二つの状態を区別できればよいわけであるから，たとえば

電圧の高，低

スイッチのON，OFF

穴のある，なし

など各種の方法により，記憶したり，伝送したり，処理したりすることができる。

**2進法**　数値を1と0で表すには，2進法（binary）を用いるのが普通である。これは

| | （参考）10進法の場合 |
|---|---|
| 整数の最下位は $2^0=1$ の位 | $10^0$ ＝一の位 |
| その左の桁は　$2^1=2$ の位 | $10^1$ ＝十の位 |
| その左の桁は　$2^2=4$ の位 | $10^2$ ＝百の位 |
| …… | …… |
| 一般に第 $n$ 桁は $2^{n-1}$ の位 | $10^{n-1}$ の位 |

という表しかたである。

（例）　2進法の1 0 1は10進法の5　$(1\times2^2+0\times2^1+1\times2^0)$

なお，小数点の右は $2^{-1}$ の桁，その右は $2^{-2}$ の桁，… となる。

**ビット**　2進法の桁数のことをビット数（bit, これは binary digit を縮めた造語）という。それは「0と1の列の長さ」といってもよい。

（例）　2進法1桁ならばビット数は1（1ビット）

0100101のビット数は7（7ビット）

**バイト**　文字は普通8箇の0と1の組合せで表す。そのようなまとまり（処理単位）をバイト（byte）という。

（1バイト）=（8ビット）

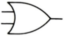
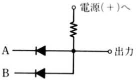
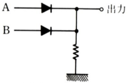
1バイトで表現できる文字の種類は $2^8=256$ である。これで英字, 数字, カナ文字, 各種の記号, などを表すことができる（下表参照）。漢字を表すにはこれでは足りないので2バイトを用いる（JIS C-6226）。

**ワード(語)**　数値は, 普通, 一定の桁数で扱われる（可変桁数で扱うことも原理的には可能であるが, 演算時間がかかるなど不利なことが多いので, 特殊な場合以外には用いられない）。このまとまり（数値処理の標準的な桁数の0と1の列）をワード（word, 語）という。ワードの長さは機種によって異なる。

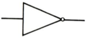
| 下位4桁 / 上位4桁 | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 | 1000 | 1001 | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0010 | 空 | ! | " | # | $ | % | & | ' | ( | ) | * | + | , | − | . | / |
| 0011 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | : | ; | < | = | > | ? |
| 0100 | @ | A | B | C | D | E | F | G | H | I | J | K | L | M | N | Ō |
| 0101 | P | Q | R | S | T | U | V | W | X | Y | Z | [ | ¥ | ] | ^ | _ |
| 1010 | | 。 | 「 | 」 | 、 | | ヲ | ァ | ィ | ゥ | ェ | ォ | ャ | ュ | ョ | ッ |
| 1011 | ー | ア | イ | ウ | エ | オ | カ | キ | ク | ケ | コ | サ | シ | ス | セ | ソ |
| 1100 | タ | チ | ツ | テ | ト | ナ | ニ | ヌ | ネ | ノ | ハ | ヒ | フ | ヘ | ホ | マ |
| 1101 | ミ | ム | メ | モ | ヤ | ユ | ヨ | ラ | リ | ル | レ | ロ | ワ | ン | ゛ | ゜ |

**文字を0と1の組合せで表すやりかたの一つの例（JIS コード）**

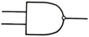
**AND と OR**　　0と1で表された情報を扱うための基本的な手段としては，AND回路とOR回路がある。いずれも，複数箇（基本的には二つ）の入力端子と1箇の出力端子をもつ。

| | AND 回路 | OR 回路 |
|---|---|---|
| 機　能 | 全部の入力端子に1が入れば出力は1，そうでなければ出力は0。 | 少なくとも一つの入力端子に1が入れば出力は1，そうでなければ出力は0。 |
| 記　号 |  |  |
| 論理式 | A∩B | A∪B |
| 回路の例 |  |  |

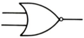
**NOT**　　基本的な論理回路としては，もう一つ，NOT 回路というのがある。これは，入力が0ならば出力は1，入力が1ならば出力は0，という回路（すなわち，0を1に，1を0に，反転する回路）である．論理式では記号¬を前につけて（たとえば¬Aというように）表す．論理回路図では，小さな白丸印で表す．



単独の NOT 回路

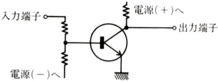


NAND，すなわち “not and” 回路



NOR，すなわち “not or” 回路

NOT 回路の例



**加算器**　2進法の加算をするには，基本的には

$0+0=0$　　$0+1=1$

$1+0=1$　　$1+1=10$　（0で，くりあがり1）

すなわち

入力が　$x=0$，$y=0$　ならば，出力は　$z=0$，$c=0$

入力が　$x=0$，$y=1$　ならば，出力は　$z=1$，$c=0$

入力が　$x=1$，$y=0$　ならば，出力は　$z=1$，$c=0$

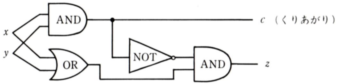
入力が　$x=1$，$y=1$　ならば，出力は　$z=0$，$c=1$

という回路を作ればよい。これらをまとめると，

$x$と$y$がともに1ならば$c=1$　（そうでないとき$c=0$）

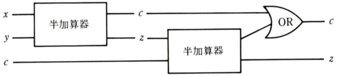
$x$か$y$が1で，$c=0$ならば，$z=1$　（それ以外は$z=0$）

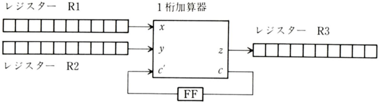
となるから，たとえば




とすればよい（ただし実際には電子回路としての諸条件を考慮して，もう少し違った形が用いられる)。

上記の機能をもつ回路を**半加算器**（half adder）という。それを2箇組み合わせて「下段からのくりあがり」を処理できるようにすると，完全な1桁加算器になる。



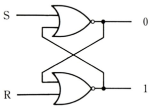



**直列加算器**　被演算数をレジスターR１，R２に置き，最下位の桁から順に１桁加算器に送って加算を行ない，結果をレジスターR３に入れる。くりあがり $c$ は，上図の FF という所で保持（一時的に記憶）しておいて，次の桁の加算の際に「下位からのくりあがり」の端子 $c'$ に送り込む。

**レジスター**(register)　処理のために数値などを一時的に置いておくための回路。記憶装置の一種とみなすこともできるが，単に記憶しておくだけでなく，**シフト**（shift，桁移動）などが可能であったり（シフトできるレジスターは**シフト・レジスター**と呼ばれる），自由なタイミングで書き込み，読み出しができるなど，処理に適した諸機能を有するのが普通である。素子としては，次に述べるフリップ・フロップを用いるのが普通である。

**フリップ・フロップ**（flip-flop，略して FF という）　端子S（set の意）に１の信号が（ごく短時間でも）入ると「０端子が０，１端子が１」という状態になり，R（reset の意）に１の信号が入ると逆の状態(０端子が１，１端子が０）になる回路。

回路の一例




（解説）

S端子に１が入ると，上の OR 回路の出力は１，それを反転した結果は０となる。そのため，下の OR 回路の出力は０，反転すると１になり，結局，０端子は０，１端子は１になる。R端子に１が入ると逆になる。

**乗算**　乗算の要領は筆算の場合とだいたい同じで，

i．$n$ 桁の被乗数に乗数の 1 桁を掛ける

ii．その結果を乗数の桁位置に合わせてシフトする

iii．それを加え合わせる

とすればよいが，2 進法の場合には「乗数の 1 桁」の値が 0 または 1 のどちらかであるから，i の乗算は事実上不要で，単に被乗数をシフトして，乗数の桁が 1 の場合だけ加え合わせればよい。

```
(例)     1 0 1 1    ←被乗数
      ×    1 0 1    ←乗数
      ------------
         1 0 1 1   }
     1 0 1 1       } これらは単に被乗数をシフトしたもの
      ------------
     1 1 0 1 1 1    ←加え合わせた結果
```

**除算**　除算の要領も筆算の場合と同様であるが，2 進法だと

商の一桁として立つのは 0 か 1 に限る

その内のどちらになるかは容易にわかる

（除数）×（商の一桁）の乗算は事実上不要

なので，結局，シフトと減算をくりかえせばよい。

```
(例)                 1 0 1      ←商
       1 0 1 1 )1 1 0 1 1 1      ←被除数
          ↑     1 0 1 1
         除数      1 0 1 1       除数を引いてゆく
                   1 0 1 1
                         0      ←余り
```

**速度とコスト**　原理は上記のとおりであるが，これを電子回路的に実現しようとすると，かなり複雑になるのでコストがかかる。そのため超小型コンピューターでは，これをプログラムで処理しているものが多い。そういう機種だと，乗算，除算は加算よりはるかに演算時間がかかる。一方，大型コンピューターでは，多数の素子を使って高速化した回路を用いているので，乗算や除算も加算とあまり変わらない時間で処理できる。

**補数**　減算を行なうには,

数値を「符号と絶対値の組」で表現しておく

減算器によって「絶対値の引き算」を行なう

という方式が考えられる（実際にこの方式を採用しているコンピューターも少なくない)。しかし，もう一つのやりかたとして

負数を「補数」という形で表現しておく

「補数の加算」によって減算を行なう

という手がある。補数というのは，負数に「非常に大きな数」を加えて（いわば，大きなゲタをはかせて）正数として扱うやりかたである。

（例）　扱う数の絶対値がすべて1より小さい場合には「非常に大きな数」として2を用いるとよい。たとえば,

$$0 \leqq a < 1 \qquad 0 \leqq b < 1$$

のとき，$-b$は，それに2を加えた値，すなわち$2-b$で表す。$a-b$の計算は,

$$a-b=a+(-b)=a+(2-b)-2$$

という形で処理する。

$$\begin{aligned}0.8-0.3&=0.8+(2-0.3)-2\\&=0.8+(1.7)-2\\&=2.5-2=0.5\end{aligned}$$

このように，負数$-x$を「足して2になる数」$2-x$で表したものを「2の補数」(2's complement）という。

補数を用いれば，減算器を用意する必要がない（加算器だけでよい)。また，加減算の際に，二つの被演算数の符号の組合せと絶対値の大小によって，絶対値を加算するか減算するかを判断する手間も省ける。そのかわり，補数を作る回路が必要になるが，2進法の場合にはこれが非常に簡単で,

1は0に，0は1に書き換える
ただし最下位だけはもとのまま

（例）　0．1011の補数は
1．0101

とすればよい。

**浮動小数点演算**　科学技術計算で扱う数値には，非常に小さな値から非常に大きな値まである。計算の際に，利用者がいちいちその位取りを調節するのではたいへんである。そこで，位取りを自動的に調節する方式が用いられるようになった。これが浮動小数点演算というやりかたであって，数値を

指数部（exponent）　位取りを表す部分

仮数部（mantissa）　有効数字を表す部分

の組として扱い，加減算の際にはまず被演算数の位取りを合わせてから演算を行なう。

（例）　10進法ならば

$$x = a \times 10^n \qquad |a| < 1$$

2 進法ならば

$$x = a \times 2^n \qquad |a| < 1$$

で表すのが普通。そのほか，2 進法を 4 ビットずつ区切って扱い

$$x = a \times 16^n \qquad |a| < 1$$

で表す方式もよく用いられている。

**2 進小数使用上の注意**　コンピューターでは無限の桁数を扱うことはできないから，無限小数は一定桁数の有限小数で代用して(近似して)扱う。その際に，どうしても丸め誤差が入る。

たとえば，1 を 3 で割り，その結果を 3 倍する，という計算は，10進法 6 桁で計算すると

1÷3＝0．333333　　（以下切り捨て）

0．333333×3＝0．999999≠1

となる。同様なことは 2 進法でも起こる。その際に注意を要するのは，

0.1　0.2　0.3　0.4　0.6　0.7　0.8　0.9

などのような「10進法の有限小数」が，2 進法だと無限小数になり，したがって正確には表現できないことが多い，ということである。

（例）　10進法の 0.1 を 2 進法で表すと

0．0001100110011001100……

になる。

## 1.3 構　　成

コンピューターは，機能的に大きく分けると，

　　入力装置　データその他を読み込む装置

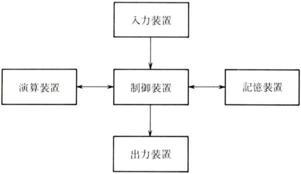
　　出力装置　結果その他を表示したり記録したりする装置

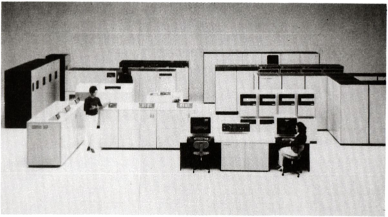
　　演算装置　演算などの処理を行なう装置

　　記憶装置　情報を記憶，保持する装置

　　制御装置　全体の制御を行なう装置

の五つの部分から成る。



大規模な構成の例

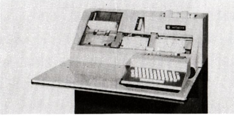
**入力手段**　現在よく使われている入力手段の種類，仕様，特長などを簡単に紹介する。

◆ **鍵盤**（keyboard）　コンピューターを対話形式で用いる場合の標準的な入力手段。またコンピューターの操作も鍵盤を介して行なうのが普通である。簡単で，安く，すぐに応答があることなどが利点であるが，手作業（それも多くの場合は素人の）であるため遅く，大量のデータの入力などには適しない。



鍵盤による入力

◆ **カード**（punched card）　縦82.32 mm，横182.55 mm，厚さ0.178 mmの厚紙に，12段80列の「穴の位置」があり，縦1列の穴の組合せで1字を表す(したがってカード1枚に80字まで記録できる)。あらかじめカード穿孔機によって記録しておいて，カード読取機にかけて入力する。読取速度は普通，1分間に数百枚～千数百枚。比較的最近まで，中～大型システムの標準的な入力手段であった。

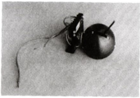


カード穿孔機

◆ **紙テープ**（punched tape）　幅25.4mmの紙テープの幅方向に8列の穴位置があり，8箇の穴の組合せで1文字を表す。読取機には，機械式の低速のもの（毎秒10字程度）と，光電式の高速のもの（毎秒数百字）がある。



文字読取機(OCR)



ジョイスティック

◆ **OCR**（optical character reader，光学式文字読取機）

◆ **OMR**（optical mark reader，光学式マーク読取機）

いずれも，手書きの情報や，誰にも読める形で印刷された情報を入力する手段である。一般の人にも親しみ易いことが最大の特長である。また，特に準備装置（カード穿孔機のような）を必要としないという利点もある。

◆ **その他**　**座標読取機**（普通は手動で，ペンやカーソルで指示した点の座標を入力する），**画像入力装置**，**AD**（analogue to digital）**変換器**，**音声入力装置**，**音響カプラー**などがある。また，制御用のコンピューターでは，バス（bus，信号母線）や直接入力用の端子を介して，デジタル情報を直接に入力することも可能である。

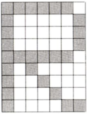
**出力手段**　出力手段には大別して

- 印刷
- 表示
- 再入力用媒体に記録
- 図化
- 直接出力

がある。その主な装置を簡単に紹介する。

◆ **プリンター**　これには，**タイプライター**(typewriter)のように1字ずつ打ってゆく低速のもの（毎秒十字ないし数十字）と，**ラインプリンター**(line printer）といって一行をまとめて（ほとんど同時に）打つ高速のもの（毎分数百行以上），および両者の中間的な性格のものがある。印字方法には，大別して

- 活字を用いる
- 点(dot）の組合せによる（右図参照）



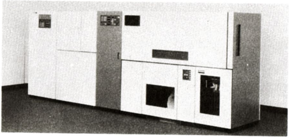
の二つがある。前者は原理的には簡単であるが文字の種類を多くするのが困難である。それに対し，後者はパターンを記憶しておく必要があるが，印字機構そのものは簡単で（したがって高速化，軽量化，コストダウンが可能），文字の種類を多くするのが容易で，やりようによってはグラフや画像を描かせることもできる，という利点がある。また，記録方法には，

a．インクリボンをたたく

b．感圧紙をたたく　　　d．インクを吹き出す

c．感熱紙に熱を加える　　e．静電的方法

などがある。この内，a，d，eは特殊な用紙を必要としないという利点がある。b，cは鮮明度がやや劣る。a，bは複写（複数枚の同時印刷）が可能。eの一種，レーザープリンターは高価であるが，きわめて高速，鮮明，汎用的である。



レーザープリンター

◆ **ディスプレイ装置**　文字だけを表示する**キャラクター・ディスプレイ**（character display）と，文字のほかに図形も表示できる**グラフィック・ディスプレイ**（graphic display）とがあり，どちらにもカラーとモノクロがある。表示にはブラウン管（cathode ray tube, 略して **CRT**）が用いられるが，

普通のテレビと同じ方式（raster scan）

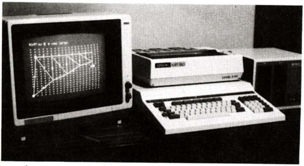
ビームを動かして線を引く方式（random scan）

の二つの方式があり，後者にはさらに

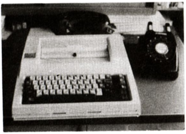
毎秒数十回くりかえし表示（refresh 型）

描いた線が消えない特殊な CRT を使う（storage 型）

の二種類がある。いずれにしても，簡単な付加装置を用いて画面のハードコピー（紙への複写）をとることができる。



**マイコンのセット**（左はディスプレイ装置，中央上はプリンター，中央下は本体とキーボード，右はフロッピーディスク装置）



ポータブル端末機

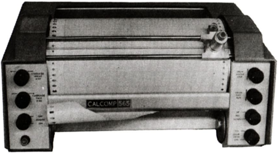
◆ **X-Yプロッター**　自動的にグラフや図面を描く装置で，

　四角い平板上をペンが動いて描く（flat bed 型）

　円筒の回転とペンの動き（1軸のみ）で描く（drum 型）

がある。紙幅25cmぐらいの小型のものから，2m×5m といった大型のものまで，また多色自動切換のできるもの，高速のものなど，各種のものがある。

◆ **直接出力**　制御用のコンピューターでは，バスや直接出力用端子を介して，直接にデジタル信号を出力することができる。またそれを **DA**（digital to analogue）**変換**して，アナログ量（たいていは電圧）の形で出力することもできる。また，**通信制御装置**と変調器を介して，通信回線に接続することも可能であり，他の機器や他のコンピューターに，直接，データを送ることができる。




X-Y プロッター

**記憶装置**　記憶装置（memory，storage）の用途には

- 処理の作業場所として使用するため　（一時的使用）
- 情報を保管しておくため　（長期間保存）

の二つがある。前者の目的には

高速に書き込み，読み出しができること

小単位（1語または1バイト単位）で出し入れできること

が必要である。そのかわり，容量は（後者にくらべれば）それほど大きくなくてもよい。一方，後者の目的には

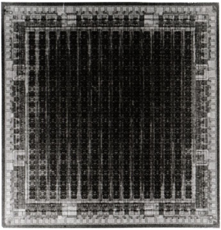
大量の情報を記録できること

が必要である。そのかわり，出し入れは小単位でなくてもよく（処理の際には一時的に高速記憶装置に移せばよい），速度も（前者にくらべれば）多少遅くてもよい。



磁気ディスク装置

◆ **集積回路（ICメモリー）**　トランジスターやダイオードは，非常に動作時間が短い。これを用いて（たとえば6ページで説明したフリップ・フロップをたくさん並べて）記憶装置を作れば，高速で，使い易い（他の素子とのつながりぐあいがよく，制御もし易い）理想的な記憶装置ができる。唯一の欠点は値段が高いことであったが，近年，大規模集積回路技術の進歩により，かなり大容量のものまで，手頃な価格で購入できるようになり，今日ではマイコンから大型コンピューターまで，ほとんどすべての機種の主記憶装置として用いられている。それでも高性能のものは高価で，少し性能を落とせば安く大容量のものを作ることができる。**バイポーラー**（bipolar）**メモリー**は，高速であるが高価なので特に高速を要する場合に用いられる。**MOS**（metal oxide semiconductor）**メモリー**は，やや低速であるが大容量のものまで比較的安く作れるので，普通はこちらが使われる。やや特殊なものに **ROM**（read only memory，読み出し専用のメモリー）がある。なお **RAM** というのは random access memory（どの番地も同じ待時間で参照できるメモリー）のこと。



IC(LSI)メモリー（実際の大きさは2 mm×2 mm）

◆ **磁気ディスク**　いわばレコードとテープレコーダーのあいのこ。形はレコードに似ている。大きさはＬＰレコードぐらい（フロッピー・ディスクはシングル盤ぐらい）。記録原理はテープレコーダーと同様，磁気による。その記録，再生ヘッドは，アームの先についていて，そのアームを移動させることにより，盤面上の位置を選択することができる。ＩＣメモリーに比較するとはるかに低速である（アームの移動に時間がかかり，また，円板が回転して所望の箇所がヘッドの所に来るまで待つ時間もかかる）が，

大量の情報を記録できる（１枚に数百ＫＢ～数十ＭＢ）

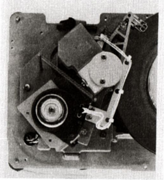
１ビットあたりのコストが安い

とりはずして保存できる（固定式のもあるが）

などの利点があるので，広く用いられている。



フロッピー・ディスク



ディスク装置の内部

◆ **磁気テープ**（magnetic tape）　コンピューター用の標準的な磁気テープは，幅12.7 mm，長さ720 m で，幅方向に９列の記録位置があり（９トラック），長さ方向１mm あたり数十ビットという高密度で０，１の情報を記録する。大量の情報を保存したり持ち運んだりするのに便利である。一方，カセット・テープは，比較的小量のデータを手軽に扱うのに適している。

◆ **磁気バブル**および**CCD**(charged coupled device)　物性を巧妙に利用した新しい記憶方法。ディスクほどの容量はないがディスクより速く，サイズが小さくて，動く部分（モーターなど）がない，などの特長がある。中程度の容量の記憶手段として利用されている。



磁気バブルメモリーの内部



MSSの内部

◆ **MSS**（mass storage system）　たくさん（数百巻〜数千巻）の磁気テープをハチの巣状のセルに入れておき，コンピューターからの指令に従って，自動的に選択，装着して記録，再生を行なう。磁気ディスクよりも遅いが，容量ははるかに大きい。

◆ **その他**　**磁気コア**(magnetic core)は，比較的最近まで，主記憶装置の標準的素子として使われていた。その前には**磁気ドラム**が主記憶装置に用いられていた。電磁遅延線，水銀タンク，ブラウン管，などが使われたこともある。しかしそれらが将来また使われる可能性は少ない。将来の可能性としては，光メモリー，超電導メモリーなどが有望視されている。

## 1.4 制御方式

**基本的な考えかた**　私達が何か複雑な仕事をする場合，

何人かで分業にして流れ作業で行なう（流れ作業方式）

最初から最後まで全部を一人で行なう（逐次作業方式）

の二つの方法がある。それと同じように，複雑な仕事をする自動機械の方式としても

単能機をたくさん並べて流れ作業で処理する

順序制御のできる汎用機1台で処理する

の二つの方法がある†。たとえば，家具を作る場合を考えてみると，前者ならば，木をけずる機械，型に合わせて切る機械，穴をあける機械，組み立てる機械，塗装する機械などを並べて，半成品を順に送ってゆく仕掛けにすればよい。これは方式として簡単であり，比較的容易に実現できるであろう。それに対し後者の方式を採用するとすれば，まず木をけずり，次に型に合わせて切り，次に穴をあけ，次に組み立てをし，最後に塗装をする——それを全部1台でやるという，ロボットのような装置になる。これを実現することは相当にむつかしいであろう。しかし，後者には前者より有利な点がいくつかある。それは

工程の変更が容易（多種小量生産が可能）

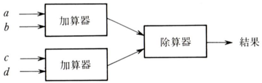
非常に複雑な工程にも対処できる

ということである。前者の方式だと，工程を変更するたびに機械を追加したり，取り除いたり，並べ替えたりする必要が起こりうる。しかし後者の方式ならば単に制御方法を変更すれば済む。また前者の方式で百工程の作業をするには原則として百台，千工程ならば千台の機械を並べなければならない。それには多額の費用がかかる。したがって，非常に複雑な仕事の場合には，後者の方がむしろ経済的になるのである。

---

† オートメーション工場のトランスファー・マシンは基本的に前者の方式，マシニング・センターは後者の方式を採用している。

**コンピューターの場合**　自動計算機の方式としても，左記と同様なことが考えられる。前者の方式ならば，たくさんの演算装置を用意しておいて，計算式に合わせてそれらをつないで，計算を行なうことになる。たとえば，$(a+b)/(c+d)$ の計算をするのであれば，次のような回路を構成して処理するわけである。




アナログ・コンピューター† は実際にそのような方式を採用している。また，近年は超高速計算機（スーパー・コンピューター）などにおいて，**パイプライン方式**という形で，このようなやりかたが部分的に使われている。

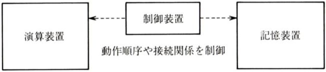
しかし，普通のコンピューターでは，後者の方式が用いられている。たとえば上の例と同じ計算 $(a+b)/(c+d)$ をするのであれば，

1） $a$ と $b$ を加える。その結果（$s$ とする）を記憶装置に入れる。

2） $c$ と $d$ を加える。その結果（$t$ とする）を記憶装置に入れる。

3） $s$ を $t$ で割る。

という形で処理する。このような方式をとれば，加算器，乗算器，…などは各1台あればよく，それらを適当な順序で用いることにより，どんな計算でもできる，という利点がある。




† 数値を電圧で表し，電圧の和，差，積，商を作る回路をつなぎ合わせて計算を行なうコンピューター。以前はデジタル・コンピューターと互角の勢力をもっていた。現在でも自動制御関係などで使われている。

**プログラム**　逐次作業方式（20ページ参照。そこでは「後者」と呼んでいる）のシステムにおける作業手順（動作順序）を記述したもののことをプログラムという（広い意味での定義）。

たとえば，前ページ中央付近に例示した1），2），3）という計算手順はプログラムである。また，たとえば自動炊飯器の

強火で加熱――100℃になったら弱火にする――20分後にOFF

というのもプログラムであり，全自動洗たく機の

吸水――2分間予洗――排水――吸水――洗剤投入――5分間洗う――排水――吸水――3分間すすぐ――排水――吸水――7分間すすぐ――排水――脱水――10分間乾燥――ブザー鳴らす

というのもプログラムである。

自動炊飯器のプログラムは変えられない（固定プログラム）のが普通であろう。全自動洗たく機には，プログラムを変えられないもの，時間だけを調節できるもの，自由に変えられるものなど，いろいろある。コンピューターのプログラムは，当然，自由に設定できなければ困る。

プログラムを機械にセットするには，どうすればよいだろう？　単純な機械ならば，ボタン操作でセットできるであろう（たとえば録音や録画用のタイマー）。しかし長い複雑なプログラムをセットするには，ボタン，ダイヤル，スイッチぐらいでは無理である。そこで，プログラムを符号化して入力する，という方法をとる。狭い意味でプログラムというのは，このための符号で表した作業手順のことである。

❋ ❋〔参考〕❋ ❋ ❋ ❋ ❋ ❋ ❋ ❋ ❋ ❋ ❋ ❋ ❋

**プログラミング**　①　計画を立案すること　②　電子計算機を使う場合に，それがどんな手順で仕事すべきかを，機械が解読できるような特別な形式（プログラム）に作りあげること。

――広辞苑（第二版）より引用

**program(me)**　**1**　プログラム，番組，（行事などの）次第書　**2**　予定，計画，計画表　**3**　（政党の）綱領　**4**　序言　**5**　（古）公示　[GK *prógramma* public notice]

――研究社の新英和大辞典より抄録

**コンピューターの場合**　演算装置の基本的な機能は電卓と同じようなものであるから，コンピューターのプログラムの内容は，だいたい「電卓で計算するときの計算手順」のようなもの，と考えてよい。

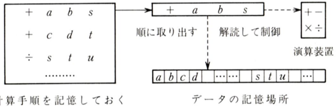



それを「符号化 (coding) する」というのは，「機械が解読し実行できるような形式で表現する」ことである。コンピューターは機械であるから，単純で規則的な処理しかできない。日本語で書かれた計算手順を解読して実行する，などということは，ちょっと無理である。もっと単純で，機械的に処理し易い形で書かれていなければ困る。

たとえば，次のような形で表現してあれば，機械的に処理することが比較的容易であろう。

| 作業の種類 | 第1被演算数 | 第2被演算数 | 結果 |
|---|---|---|---|
| + | $a$ | $b$ | $s$ |
| + | $c$ | $d$ | $t$ |
| ÷ | $s$ | $t$ | $u$ |

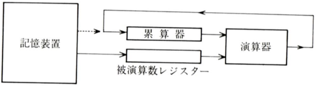
このような表を「記憶」させておいて，1ステップずつ順に実行させれば，計算ができるわけである。



**1アドレス方式**　機械処理の作業手順表の形式はいろいろある。前ページに例示した形は「3アドレス方式」と呼ばれている（各ステップで，被演算数2箇と結果1箇，計3箇の記憶場所を指定するから）。それに対し，次のような表しかたも考えられる。こちらは，各ステップで指示される記憶場所が1箇だけなので「1アドレス方式」と呼ばれている。

| 作業の種類 | 作業の対象 | （意　味） |
|---|---|---|
| LOAD | $a$ | $a$を（累算器に）もってこい |
| ADD | $b$ | （その結果に）$b$を加算せよ |
| STORE | $s$ | （その結果を）$s$に格納せよ |
| LOAD | $c$ | $c$を（累算器に）もってこい |
| ADD | $d$ | （その結果に）$d$を加算せよ |
| STORE | $t$ | （その結果を）$t$に格納せよ |
| LOAD | $s$ | $s$を（累算器に）もってこい |
| DIVIDE | $t$ | （その結果を）$t$で除算せよ |
| STORE | $u$ | （その結果を）$u$に格納せよ |

ここで**累算器**（accumulator）というのは，演算結果が入るレジスターで，それがすぐ次の演算の被演算数レジスターとして使用できるものをいう（要するに電卓の表示窓に対応するレジスター）。




このような方式は，合計や連乗の計算に便利である。

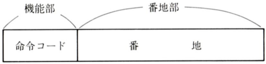
（例）$a+b+c=s$の計算

| 作業の種類 | 作業の対象 |
|---|---|
| LOAD | $a$ |
| ADD | $b$ |
| ADD | $c$ |
| STORE | $s$ |

（例）$a\times b\times c=t$の計算

| 作業の種類 | 作業の対象 |
|---|---|
| LOAD | $a$ |
| MULT. | $b$ |
| MULT. | $c$ |
| STORE | $t$ |

**命令の構成**　左記のような作業手順表の各ステップは，**命令**（コンピューターに対する命令，という意味。instruction）と呼ばれ，一定の形式で表現され，記憶される。1アドレス方式の場合についていえば，だいたい次のような形式をとる。




（例）　1 0 1 1 0 1 0 0 0 1 0 0 1 0 1 1 1 1 0 1 1 0

命令コード（instruction code, operation code）は作業の種類を符号で表したもので，記憶装置の読み出し，書き込み，整数および浮動小数点数の加減乗除，シフト（桁移動），数値や文字列の比較，入出力装置の制御など，百種類ぐらいあるのが普通である。

番地（address）は記憶装置の中の位置（どこから読み出すか，どこに書き込むか）を示すもので，0番から始まる一連番号で指示するのが普通である。

**実際のしくみ**　コンピューターの内部のしくみは，原理的には以上のとおりであり，初期の時代には実際このような構造のコンピューターが使われていた。しかし近年は，処理を高速化するため，また，大規模な問題の処理を容易にするため，

　　複数箇の累算器を使用する
　　番地指定方式を多様化する
　　命令の形式も多様化する

など，各種の工夫が加えられ，実際の構造はかなり複雑なものになっている。一方，マイクロ・コンピューター（特に4ビットや8ビットのもの）は，情報を語単位でなく文字単位で扱うので，上記とはかなり違った構造になっている。コンピューターの専門家になるのには，それらについての詳しい知識が必要であるが，一般の利用者は上記の程度のことを常識として知っていれば十分であろう。なぜなら，普通の仕事のプログラムは，これから説明する「プログラミング言語」で書けばよいからである。

## 1.5　プログラミング言語

プログラムを作成するには，

　　プログラムをまず自然言語（私達の場合なら日本語）で書く

　　次にそれを機械語（前ページで述べたような形）になおす

というのが一つのやりかたである。常識的に考えると，これでべつに不都合はなさそうである。実際，コンピューターの発展の初期の時代には，このような方法でプログラムを作っていた。

しかし，複雑なプログラムをこういう方法で作るのは非常にたいへんである。第一に問題になるのは，自然言語で書かれたプログラムを機械語に書きなおす（符号化する）作業量である。長いプログラムだと，数千行～数万行に及ぶものがある。取り扱うデータの箇数も，数万～数十万になることがある。その番地を割り当てたり，複数箇のレジスターの活用法を考えたりしながら，命令の列の形になおしてゆくのには多大の労力を必要とする。

それでも，たいへんな手間をかけて作ったプログラムが正常に動いてくれれば，まだよい。もしも正常に動かなかったらどうするか？　長大な命令の列を調べて原因を見つけ出し修正するのが，これまたたいへんである。また，自然言語にはあいまいなところがあるから，「自然言語で書かれたプログラム」は，解釈のしようによって，正しいプログラムになったり正しくないプログラムになったりすることがありうる。

そういう問題点を一挙に解決するために，

　　プログラムを記述するための人工言語を作る

　　利用者はその言語によってプログラムを書く

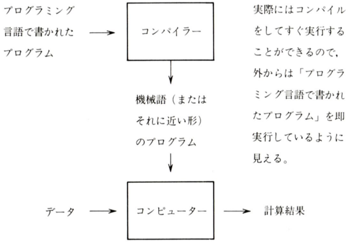
　　その言語から機械語への翻訳はコンピューターで行なう

という方式が開発された。この「プログラムを記述するための言語」のことを，プログラミング言語（programming language）という。今日では，大部分のプログラムが，このような方式によって作成されている。

代表的なプログラミング言語としては，次のようなものがある。

| 名　称 | 寸　評 |
|---|---|
| BASIC | 簡単で覚え易い |
| FORTRAN | 科学技術計算用 |
| COBOL | 事務処理用 |
| PL／I | 万能言語 |
| ALGOL | 言語の理想を追求したもの |
| PASCAL | ALGOL を単純化，実用化したもの |
| LISP | 人工知能などの高度な情報処理用 |
| SNOBOL | 文字列処理用 |
| APL | コンパクトに書ける |
| ADA | 機能強大 |

**コンパイラー**　　プログラミング言語で書かれたプログラムを，機械語（またはそれに近い形）に変換する作業を**コンパイル**（compile），それを行なう手段（コンピューターで翻訳を行なうためのプログラム）をコンパイラー（compiler）という。



## 1.6　オペレーティング・システム (OS)

昔（初期の時代）のコンピューターの利用形態は，非常に単純で常識的であった。要するに，一度に一つの仕事しかしなかった。たとえば

まず，前の人の仕事の終るのを待つ

自分の番が来たら，コンピューターの状態を白紙にもどし

プログラムを入力し

データを入力し

計算をし

（やがて結果が印刷される）

この間，二つのことを同時にはしない。たとえば入力中，出力中は計算は休止。

終ったら次の人にコンピューターを引き渡す

というぐあいである。当時は処理速度が遅くて，ちょっとした計算にも数十分かかり，また，利用者が比較的少なかったので，このような使いかたでよかった。しかし，その後，コンピューターが速くなって，簡単な仕事ならば数秒で終るようになり，同時に，利用人口および仕事の件数が急増したので，コンピューターをもっと能率よく運転することが必要になってきた。また，大容量の記憶装置が普及したので，よく使うプログラムやデータは記憶装置に常時入れておくようになった。そこで

コンピューターを皆で一緒に使う

同時にできることがあれば，なるべく同時に行なう

それらの管理をコンピューターが行なう

という方式が開発された。これがオペレーティング・システム（ＯＳ，operating system）である。これにも長い歴史があって，最初は単純な「自動運転システム」であったものが，しだいに機能を追加されて複雑なしくみになったのであるが，今日のＯＳは，だいたい次のような機能をもっている。

◆ **記憶場所の管理**　記憶装置を複数の人が分けて使うのであるから，誰はどの場所，という割り当てを行ない，常にその場所を使うように，また他人に迷惑をかけないように，管理する。

◆ **入出力の管理** 現在使われている大部分のコンピューターにおいては，入力，処理（計算），出力の機構が独立していて

i．プログラムやデータを入力して記憶装置に入れる

ii．記憶装置に入っているプログラムを実行し，データを処理して結果を記憶装置に入れる

iii．記憶装置に入っている内容を出力する

を同時に（平行して）行なうことができる。そこで，ある人の処理をしながら，同時に，あとの人のプログラムやデータを入力して記憶装置の一部（いわば待合室）に入れておき，同時に，既に計算が済んで出力待ちの形で記憶装置に入っている結果を出力することができる。このような扱いをするための業務（いわば受付窓口業務）を行なう。また，入出力に異常があればその対策をとる（たとえばプリンターの紙がなくなればオペレーターに警告）。

◆ **実行管理** 一つの計算を行なうにも，

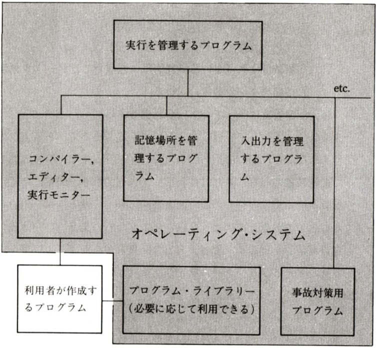
入力　コンパイル　編集†　実行　出力

というようにいくつもの段階がある。これらが連続して円滑に行なわれるように管理する。予定時間を超過したり，異常（たとえば0による割り算）がたびたび起こるようなら仕事を打ち切るなどの監視も行なう。また，非常に長時間かかる計算を中断して，短い計算を優先処理するとか，ある仕事の途中でディスクからの入力待ちになったとき，待っている間に他のプログラムを実行させるなど，きめの細かい管理を行なう。

◆ **その他** 利用資格のチェック，使用料金の計算，運転日誌や運転統計の作成，遠方にある端末機や他のコンピューターとの間の通信の制御，各種の割り込みの処理，などを行なう。

---

† **編集**する（edit）とは，いくつかのプログラムをつなぎ合わせること。既存プログラムの組み込みや，分割して作成したプログラムの結合などに必要。




オペレーティング・システム概念図

私達はハードウェアを直接に使うのではなくて

ハードウェア　＋　オペレーティング・システム

（機能は単純）　　（高度な機能をもつ）

という複合体を使うわけである。

**ジョブ制御言語**(job control language, JCL) 利用者がオペレーティング・システムに情報を伝えるための言語。膨大な仕様で全部はとても覚えきれないが，普通は，利用登録，言語指定，実行方法の指定，ファイルの利用法，ライブラリーの呼び出しかた，などを知っていれば十分である。その書きかたは，機種によって，また運営方法によって異なるが，だいたい次のような順序になっている。

1) **JOB制御文** 先頭に置く。利用者番号，利用目的(課金コード)，ジョブクラスの指定，出力方法の指定，打ち切り時間の指定などを行なう。いわば，利用開始手続きである。

2) **ファイル制御文** ディスク・ファイルなどを利用する場合，そのファイル名などを登録する。

3) **プログラミング言語制御文** 使用するプログラミング言語の種類や，処理に関するオプション(選択できる仕様)の指定などを行なう。これに続いて，ソース・プログラムを入力する。

4) **編集制御文** ライブラリー・プログラムその他，既にディスクに入っているプログラムを組み込んだり修正したりする。

5) **実行制御文** 実行開始を指示する。これに続いて，データを入力する。

6) **終端の印** 当該ジョブの入力の終端であることを示す。

**私達が使う JCL**

## 1.7 小　　史

1642　Pascalが手まわし式の加算機を発明。計算を機械で行なうことの可能性を示した。

1822　Babbageがプログラム制御による機械式汎用自動計算機の研究に着手。以後約50年にわたり試作，改良を続ける。

1944　リレー式計算機ハーバードMARK-Ⅰ完成。電気的論理回路による汎用自動計算機の方式を確立した。

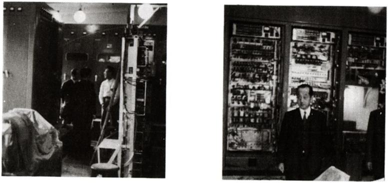
1946　電子計算機ENIACが完成。電子回路による計算の可能性と高速性を実証。以後10年間にわたり実用に使われる。

1949　プログラム内蔵方式のコンピューターEDSACが完成。基本的なソフトウェア技法が開発される。

1950年代　コンピューターが商品化されて広く使われるようになる。わが国でも1955年から1957年にかけて，リレー式，真空管式，パラメトロン式，トランジスター式の計算機があいついで完成。

1960年代　プログラム言語，オペレーティング・システム，遠隔利用，記号処理，パターン認識など，高度な利用技術が進む。

1970年代　集積回路技術の進歩が著しいコスト低下をもたらし，利用分野が急激に広がる。



開発中のTACと山下英男先生（1958年）

# 2 BASIC

## 2.1 BASICとは

BASICは教育用に開発されたプログラミング言語である。簡単で覚え易く，そのわりにかなりのことができるので広く愛用されている。

| | |
|---|---|
| 名前の由来 | Beginner's All-purpose Symbolic Instruction Code |
| 開発した人 | J. G. Kemeny と T. E. Kurtz (Dartmouth 大学) |
| 完成した年 | 1964年に第1版が出た。その後，数回，機能の拡張が行なわれている。 |

仕様は機種によってかなり異なる。大別して次の3種類がある。

簡略BASIC　無くても済む機能をなるべく省き，基本的な部分だけを残したもの。

基本BASIC　国際標準規格† の**基本BASIC**相当のもの。DATA, DEF, DIM, END, FOR, GOSUB, GOTO, IF, INPUT, LET, NEXT, ON, OPTION, PRINT, RANDOMIZE, REM, RESTORE, RETURN, STOP の各文を使用できる。

拡張BASIC　各種の機能を大幅に追加したもの。

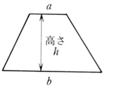
本書では，基本BASICを中心に説明する。

---

† 1979年に決定された規格案。JISもほぼ同じ内容になる見込み。

## 2.2 簡単な計算と入出力

### 2.2.1 台形の面積




右の図のような台形の面積を計算するプログラムを作ってみよう。

〔**設計**〕 計算式は

S＝( $a+b$ ) $h/2$

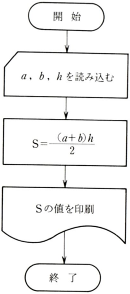
処理の手順は

1） データ $a$, $b$, $h$ を読み込む。

2） 面積Sを計算する。

3） 答を印刷する。

とすればよいであろう。手順を**流れ図**（flow chart）で表すと右図のようになる。




〔**製作**〕 BASICでは上記の手順1）を次のように表す。

```
10 INPUT A,B,H
```

行番号　「入力せよ」という意味　入力する変数の名前

手順2）は次のように表す。

```
20 LET S=(A+B)*H/2
```

行番号　「代入せよ」という意味　代入先　計算式

手順3）は次のように表す（形式は手順1）とほぼ共通)。

```
30 PRINT S
```

最後に実行の終りを表すEND文を書く。

```
40 END
```

**BASICの文法―(1)―式と代入文の書きかた**

◆ すべての文に行番号を付ける。

行番号には1以上9999以下の整数が使用できる。

プログラムは行番号の順に書くものとする。

文は原則として行番号の順に実行される。

◆ 変数名としては，

英字1字　　または　　(例)　H

英字1字の次に数字1字を付けたもの　(例)　F1

が使用できる†。

◆ LET文(代入文)は次の形式で書く†。

行番号　LET　変数名＝式

＝の左は計算結果の代入先の変数名，＝の右は計算式である。

◆ 計算式を書くときの注意。

乗算記号は＊印を用いる。　(×印や・印は使えない)

乗算記号の省略は許されない。

除算記号は／印を用いる。　(÷印は使えない)

カッコは(　)のみ。ただし何重にも使える。

(悪い例)　A・B　C×D　EF　G÷H

(良い例)　A＊B　C＊D　E＊F　G／H

$\frac{c}{ab}$ を計算したい場合 { (悪い例)　C／A＊B ; (良い例)　C／(A＊B) }

**演習**　1　誤りを正せ。

(1)　−3C　(2)　(A+B)C　(3)　A・B

2　BASICの式になおせ。

(1)　$\frac{1}{x+y}$　(2)　$\frac{1}{xy}$　(3)　$\frac{y}{2x}$

(4)　$\frac{b}{a}+\frac{d}{c}$　(5)　$\frac{a+b}{\alpha+\beta}$

† 3字以上の変数名を許したり，LETの省略を許す機種もある。

〔できあがったプログラム〕

```
10 INPUT A,B,H
20 LET S=(A+B)*H/2
30 PRINT S
40 END
```

〔別解1〕 A, B, Hの値をプログラムの中で直接に式で書いてもよい。1〜2ケースだけの計算ならば，この方が簡単，明快である。

```
10 LET A=3
11 LET B=8
12 LET H=4.5
20 LET S=(A+B)*H/2
30 PRINT S
40 END
```

〔別解2〕 PRINT文に式を書いてもよいので，次のように3行で書くことができる。

```
10 INPUT A,B,H
30 PRINT (A+B)*H/2
40 END
```

〔別解3〕 プログラムに注釈を付記したり，結果に見出しを付けて出力することができる。

```
 5 REM ---DAIKEI NŌ MENSEKI---
10 INPUT A,B,H
20 LET S=(A+B)*H/2
31 PRINT "DAIKEI NŌ MENSEKI"
32 PRINT "A=";A
33 PRINT "B=";B
34 PRINT "H=";H
35 PRINT "S=";S
40 END
```

**BASICの文法—(2)—PRINT文とREM文**

◆ PRINT の右に書けるのは

| | | |
|---|---|---|
| 変数名 | (例) | R |
| 式 | (例) | 2＊L |
| 2重引用符で囲んだ文字列 | (例) | "YMŌ" |

それらをコンマで区切って並べたもの

(例)　A，B，X，A＊X＋B，"CM"

などである。式が書いてあると，PRINT 文の実行の時点で，その式の値を計算し出力する。2重引用符で囲まれた文字列は，そのまま(文字列が)出力される。

◆ 原則として PRINT 文一つが出力の1行に対応する。

(例)　PRINT A，B　と書けば横に並べて出力される。

(例)　PRINT A
PRINT B　と書けば縦に並べて出力される。

ただし，PRINT 文の最後がコンマまたはセミコロンで終っていれば，改行されず，次の PRINT 文では，いま出力したものの右に続けて出力される。

(例)　PRINT A，
PRINT B　と書けば横に並べて出力される。

◆ 印字項目の区切り記号はコンマとセミコロンの2種類がある。セミコロンを用いると「前の項目にすぐ続けて次の項目を印字せよ」という意味になり，コンマを用いると「各項目を一定の長さ(普通15字前後)で出力せよ」という意味になる(ただし両者を混用した場合の規則はもう少し複雑である)。

◆ REM 文はプログラムの中に注釈(人間のための覚え書き)を付記するためのもので，実行上の効果は無い。注釈には，英字，数字および記号＋－＊／．，：；()<>^＝_'？！＃¥％＆ などが使用できる。なお REM は remark の意味。

### 2.2.2 組み込み関数

正の数 $x$ と角度 $\theta$（単位は度）を読み込み

$$y_1=\sqrt{1+\sqrt{x}}+\sqrt[3]{x}$$

$$y_2=\log_{10} x$$

$$y_3=e^{-x}\cos\theta$$

の値を計算して出力するプログラムを作ってみよう。

**〔設計〕** BASIC で使用できる関数の種類は機種によって異なるが，右のページに挙げた関数は，だいたいどの機種でも使用できる。これだけを用いて上記の計算をするには次のように変形して扱えばよい。

$$\sqrt[3]{x}=x^{(1/3)}$$

$$\log_{10} x=\log_e x/\log_e 10$$

$$\theta\text{度}=(\pi/180)\,\theta\text{ラジアン}$$

**〔プログラム例〕** $\theta$ を変数名Tで表すことにする。

```
10 INPUT X,T
20 LET Y1=SQR(1+SQR(X))+X^(1/3)
30 LET Y2=LOG(X)/LOG(10)
40 LET C=3.1416/180
50 LET Y3=EXP(-X)*COS(C*T)
60 PRINT "Y1=";Y1
70 PRINT "Y2=";Y2
80 PRINT "Y3=";Y3
90 END
```

**〔注意と補足〕**

1） 三角関数の角度の単位はラジアンである。

2） LOG（X）は自然対数である。

3） ATN（X）は $-\pi/2$ から $\pi/2$ の間の値をとる。

4） （ ）の中に式を書いてよい。

5） $x\geqq 0$ ならば，INT（X）は $x$ の整数部分に等しい。

**BASICの文法―(3)―関数の書きかた**

◆ $x^y$ の表しかたは機種によって異なり，

X^Y　　X↑Y　　X**Y

などと書く。Yは必ずしも整数でなくてもよい。

◆ 次のような関数が使用できる。

| | | | |
|---|---|---|---|
| SQR(X) | 平方根 | $\sqrt{x}$ | |
| EXP(X) | 指数関数 | $e^x$ | |
| LŌG(X) | 自然対数† | $\log_e x$ | |
| SIN(X) | 正弦 | $\sin x$ | 単位はラジアン |
| CŌS(X) | 余弦 | $\cos x$ | 単位はラジアン |
| TAN(X) | 正接 | $\tan x$ | 単位はラジアン |
| ATN(X) | 逆正接 | $\arctan x$ | 単位はラジアン |
| ABS(X) | 絶対値 | $\lvert x \rvert$ | |
| SGN(X) | 符号 | $=\begin{cases} -1 & (x<0\text{のとき}) \\ 0 & (x=0\text{のとき}) \\ +1 & (x>0\text{のとき}) \end{cases}$ | |
| INT(X) | 整数部 | $x$を越えない最大の整数 | |
| RND | 乱数†† | 0と1の間の一様乱数 | |

**演習**　**1**　BASICの式になおせ

(1) $\sqrt{1-x^2}$　(2) $\log\log x$　(3) $e^{2x}$

(4) $\sin 10°$　(5) $|x|-\arctan y$

**2**　正の整数 $n$ と正数 $a$ を読み込み，一辺の長さが $a$ の正 $n$ 角形の面積

$$S=\left(\frac{a}{2}\right)^2\frac{n}{\tan(\pi/n)}$$

を計算して出力するプログラムを作れ。

---

† MZ-80ではLŌGが常用対数を表すので注意を要する。

†† 機種によっては，RND(X)と書く。Xは単なる形式的引数のこともあり，出発値設定方法の指示に用いられることもある。

### 2.2.3 応 用 問 題

**1 (和差積商)**

二つの数$a$，$b$を入力し，その和，差，積，商を計算して出力するプログラムを作れ。

**2 (単位換算)**

次の諸量の単位換算を行なうプログラム（一方の単位で表された値を入力し他方の単位に換算して出力する）を作れ。

i ）セ氏温度C ↔ カ氏温度F

（公式　C＝(５/９)・(F－32)）

ii ）インチ ↔ mm

（公式　1インチ＝25.4 mm）

iii ）ポンド ↔ kg

（公式　1ポンド≒0.45359 kg）

iv ）馬力 ↔ ワット

（公式　1馬力＝736ワット）

v ）気圧 ↔ ミリバール ↔ mm 水銀柱

（公式　1気圧＝1013.25ミリバール＝760 mmHg）

vi ）秒速m ↔ 時速 km

（公式　秒速1 m＝時速 3.6 km）

vii ）ラジアン ↔ 度

（公式　1ラジアン＝180/$\pi$度）

**3 (指標値)**

次の指標値を計算するプログラムを作れ。

i ）偏差値

（公式　偏差値＝((点数－平均点)/標準偏差)×10＋50）

ii ）不快指数

（公式　不快指数＝0.99×最高気温＋0.36×平均露点温度＋41.5）

iii ）デシベル

（公式　1 db＝20 $\log_{10}$(当該値/基準値)）

**4　（面積，体積など）**

次の値を計算するプログラムを作れ。

i）底辺 $a$，高さ $h$ の三角形の面積

ii）半径 $r$ の円の面積

iii）半径 $r$ の球の体積および表面積

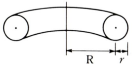
（公式　$V=(4/3)\pi r^3$　$S=4\pi r^2$）

iv）底面の半径 $r$，高さ $h$ の円錐(すい)の体積

（公式　$V=(\pi/3)r^2h$）

v）輪環面（torus，ドーナツ形）の表面積

（公式　$S=4\pi^2Rr$）




vi）主軸の長さ $2a$，$2b$，$2c$ のだ円体の体積

（公式　$V=(4/3)\pi abc$）

**5　（精度を調べる）**

次の値をコンピューターで計算し，厳密な値と比較せよ(厳密な値の，小数点以下 12 桁までを下記カッコ内に示す)。

$\sqrt{2}$　（1.414213562373）

$\log_e 2$　（0.693147180560）

$e^1$　（2.718281828459）

$\sin 10°$　（0.173648177667）

arctan 1（0.785398163397）

**6　（三角関数）**

$\alpha$ と $\beta$ の値を入力し，次の恒等式の左辺と右辺を別々に計算し，それぞれの値とその差を出力するプログラムを作れ。

$$\sin(\alpha+\beta)=\sin\alpha\cos\beta+\cos\alpha\sin\beta$$

$$\cos(\alpha+\beta)=\cos\alpha\cos\beta-\sin\alpha\sin\beta$$

$$\tan(\alpha+\beta)=\frac{\tan\alpha+\tan\beta}{1-\tan\alpha\tan\beta}$$

## 2.3　くりかえし

### 2.3.1　数表を作る

1から10までの整数の，2乗，逆数，平方根の数表を作ってみよう。

〔設計〕　Nの値を1から10まで変えて

N　　$N^2$　　1/N　　$\sqrt{N}$

の値を印刷すればよい。

〔製作〕　BASICでは，これを次のように書く。

```
10  FŌR  N=1  TŌ  10
↑   └────────────────┘
行番号      「Nの値を1から10まで変えて
              以下のことを実行せよ」という意味

20  PRINT  N,N*N,1/N,SQR(N)
↑   ↑      ↑  ↑   ↑    ↑
行番号 |     N  N²  1/N   √N
「印刷せよ」 └──────────────────┘
という意味        印刷すべき内容

30  NEXT  N
行番号      「Nを次の値に変えて，FORの次の文にもどれ」
            という意味。反復部分の終端を示す。
              反復が終ると次の文に進む。

40  END
```

〔できあがったプログラム〕

```
10  FŌR  N=1  TŌ  10
20     PRINT  N,N*N,1/N,SQR(N)
30  NEXT  N
40  END
```

**BASICの文法—(4)—FOR文, NEXT文**

◆ FOR文の書きかたは

　　文番号　FŌR　変数名＝始値　TŌ　終値

である。その右に「STEP　増分」を付けてもよい。

(例)　　7　FŌR　I＝2　TŌ　8　STEP　3

増分を書かなければ増分1となる。

◆ 始値, 終値, 増分としては, 定数, 変数はもちろんのこと, 一般に算術式を書くことも許されている。その値は0や負数でもよい。また, 必ずしも整数でなくてもよい。

(例)　　1　FŌR　A＝0　TŌ　−3　STEP　−1

　　　25　FŌR　C＝D　TŌ　E＋F

　　　60　FŌR　H＝2　TŌ　3　STEP　0.5

ただし, 0.1などを増分として用いると, 丸め誤差のために反復回数が思いどおりにならない場合があるので注意すること。

◆ NEXT文の書きかたは

　　文番号　NEXT　変数名

〔出力例〕

| N | $N^2$ | 1/N | $\sqrt{N}$ |
|---|---|---|---|
| 1 | 1 | 1 | 1 |
| 2 | 4 | .5 | 1.41421 |
| 3 | 9 | .333333 | 1.73205 |
| 4 | 16 | .25 | 2 |
| 5 | 25 | .2 | 2.23607 |
| 6 | 36 | .166667 | 2.44949 |
| 7 | 49 | .142857 | 2.64575 |
| 8 | 64 | .125 | 2.82843 |
| 9 | 81 | .111111 | 3 |
| 10 | 100 | .1 | 3.16228 |

**演習**　1　$x$の値を1から0.1おきに2まで変えて,

$$y=2x^2+3x-1$$

の値を計算し印刷するプログラムを作れ。

2　$a$, $b$, $c$, $x_1$, $x_2$, $\Delta x$の値を読み込み, $x$の値を$x_1$から$x_2$まで$\Delta x$おきに変えて$y=ax^2+bx+c$の値を計算し印刷するプログラムを作れ。

### 2.3.2 級数の和

級数

$$1+\frac{1}{1!}+\frac{1}{2!}+\frac{1}{3!}+\cdots+\frac{1}{10!}$$

の和を計算するプログラムを作ってみよう。

〔設計〕 $k$ の値を 0 から 1 0 まで変えて，第 $k$ 項 $a_k=1/k!$ の値を計算し，合計すればよいであろう。$a_k$ の値を計算するには，$k!$ の値を求めて逆数をとる，という方法も考えられるが，

$$a_k=\frac{1}{k!}=\frac{1}{k}\cdot\frac{1}{(k-1)!}=\frac{1}{k}\cdot a_{k-1}$$

という関係を用いて，

$$a_0=1$$

$$a_k=a_{k-1}/k \qquad (k=1,\ 2,\ \cdots,\ 10)$$

の形で計算する方が簡単である。その合計を求めるには，第 $k$ 項までの部分和を $S_k$ として，やはり漸化式の形

$$S_0=1$$

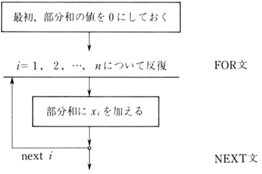
$$S_k=S_{k-1}+a_k \qquad (k=1,\ 2,\ \cdots,\ 10)$$

で計算することができる。なお，$a_k$，$S_k$ が計算されれば，以後は，$a_{k-1}$，$S_{k-1}$ は必要ないから，それを消して同じ場所に書き込むことにすると，記憶場所は一般項の部分和のために各 1 語あればよい。

〔プログラム例〕

```
10 LET A=1                  一般項，最初は 1
20 LET S=1                  部分和，最初は 1
30 FŌR K=1 TŌ 10
40    LET A=A/K             a_k の計算
50    LET S=S+A             部分和に加え込む
60 NEXT K
70 PRINT S                  結果の印刷
80 END
```

**〔総和Σの計算法〕** 一般に，$\sum_{i=1}^{n} x_i$ は次のようにして計算できる。




（終了時の部分和の値が $\sum_{i=1}^{n} x_i$ である。）

**演習**　**1**　左記のプログラムを改造して，中間結果 $a_k$，$S_k$（$k$＝0，1，…，9）を印刷できるようにせよ。

**2**　左記のプログラムを改造して，

$$1+\frac{x}{1\,!}+\frac{x^2}{2\,!}+\frac{x^3}{3\,!}+\cdots+\frac{x^{10}}{10\,!}$$

の値を計算できるようにせよ（$x$ の値は最初に INPUT 文で読み込む）。

**3**　$x$ および $n$ の値を読み込み，次の級数の和を計算し印刷するプログラムを作れ。

(1)　$1-\dfrac{x^2}{2\,!}+\dfrac{x^4}{4\,!}-\cdots+(-1)^n\dfrac{x^{2n}}{(2n)!}$

(2)　$x-\dfrac{x^3}{3}+\dfrac{x^5}{5}-\cdots+(-1)^{n-1}\dfrac{x^{2n-1}}{2n-1}$

(3)　$\sin x-\dfrac{\sin 2x}{2}+\cdots+(-1)^{n-1}\dfrac{\sin nx}{n}$

### 2.3.3 応用問題

**1 （数表を作る）**

次の数表を出力するプログラムを作れ。

i ）$\theta=0^\circ$ から $90^\circ$ まで $10^\circ$ おきの

$$\sin\theta \qquad \cos\theta \qquad \tan\theta$$

の数表。

ii ）$x=0.1$ から $2.0$ まで $0.1$ おきの常用対数 $\log_{10}x$ の数表。

iii ）$n=1$ から 10 までの $n!$ の数表。

**2 （料金早見表）**

ガソリン・スタンド用の料金早見表を作りたい。単価を 4 種類（製品の種類などによって異なる。例えば，リッターあたり 138 円，145 円，155 円，170 円）読み込み，1 リッターから0.1リッターおきに30リッターまでの料金表を出力するプログラムを作れ。

**3 （複利計算）**

元金 1 万円を複利で運用したら 1 年後，2 年後，…，20年後にいくらになるか（元利合計）の表を作りたい。3 段階の年利率（例えば，3.5 %，5 %，6 %）を読み込み，上記の表を出力するプログラムを作れ。

**4 （積立預金）**

毎年 1 万円を積立預金したら，1 年後，2 年後，…，20年後の元利合計がいくらになるかについて，前問と同様の表を出力するプログラムを作れ。

**5 （年金）**

退職金1000万円を原資とし，毎年100万円ずつ引き出して使用したとして，10年後までの毎年の元利合計を計算するプログラムを作れ。ただし預入れ時点には引き出さず，1 年後から引き出すものとする。また，利率は前記**3** と同様に 3 段階の値を指定できるようにする。

**6 (極限値)**

数値計算によって数列の極限値の厳密な値を求めることはできないが，極限値に近い値をいろいろ計算してみることによって極限値を推測することはできる。たとえば

$$a=\lim_{x\to 0} x^x$$

の値を知るには，$x_n=10^{-n}$ に対する $a_n=x_n^{x_n}$ の値を $n=1$，2，3，… に対して計算してみればよいであろう。

i）$n=1$ から10までに対し，上記 $a_n$ の値を計算して出力するプログラムを作れ。

ii）$a=\lim_{x\to 0}\dfrac{\sin x}{x}$ について同様なプログラムを作れ。

iii）$a=\lim_{n\to\infty}\left(1+\dfrac{1}{n}\right)^n$ について同様なプログラムを作れ。

iv）半径1の円に内接する正 $n$ 角形の周の長さ $a_n$ と，外接する正 $n$ 角形の周の長さ $b_n$ は次式で計算できる。

$$a_n=2n\sin\frac{\pi}{n} \qquad b_n=2n\tan\frac{\pi}{n}$$

$n\to\infty$ とするときの，これらの値の極限値を知るために，上と同様なプログラムを作れ。

v）$\tan x$ を連分数に展開すると次のようになる。

$$\tan x=\cfrac{x}{1-\cfrac{x^2}{3-\cfrac{x^2}{5-\cfrac{x^2}{7-\cfrac{x^2}{9-\text{以下同様}}}}}}$$

$x$ の値を入力し，上記の式（適当なところまで）で $\tan x$ の値を近似計算するプログラムを作れ。

## 2.4 添字付き変数

### 2.4.1 平均値と標準偏差

20箇のデータ $x_1$, $x_2$, …, $x_{20}$ を読み込み，その平均値と標準偏差を計算して印刷するプログラムを作ってみよう。

**〔計算式〕** 一般に，$n$ 箇のデータ $x_1$, $x_2$, …, $x_n$ の平均値 $\mu$ および標準偏差 $\sigma$ の計算式は，

$$\mu=\frac{1}{n}\sum_{i=1}^{n}x_i \qquad \sigma=\sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i-\mu)^2}$$

**〔プログラムの設計〕** 全部のデータ $x_1$, $x_2$, …, $x_n$ を記憶装置の中に**配列**(表)の形で入れて，**添字付き変数**の形で扱うことにする。手順は

1) データの格納場所（配列）を用意する。
2) $n$ およびデータ $x_1$, $x_2$, …, $x_n$ を読み込む。
3) $\sum x_i$ を計算する。
4) それを $n$ で割る（その結果が $\mu$）。
5) $\sum(x_i-\mu)^2$ を計算する。
6) それを $n$ で割り，その平方根を求める（その結果が $\sigma$）。
7) 結果を印刷する。

**〔プログラム例〕**

```
 10 DIM X(100)
 20 INPUT N                          nを読む
 30 FOR I=1 TO N                  ┐
 40    INPUT X(I)                 │  データを読む
 50 NEXT I                        ┘
 60 LET W=0                       ┐
 70 FOR I=1 TO N                  │
 80    LET W=W+X(I)               │  Σxiの計算
 90 NEXT I                        ┘
100 LET M=W/N                        平均値μ
```

**BASICの文法―(5)―配列と添字**

◆ 配列を使用するときは，原則として†，プログラムの冒頭で

DIM 配列名(添字の上限)

の形で宣言しておく必要がある（DIMはdimensionの意）。

◆ 配列名は英字1字に限る。††

◆ 配列要素は，

配列名(添字)

の形で引用することができる。

◆ 添字の値として，負の数や，DIM文で宣言した上限を越える値を用いてはならない。

◆ 添字の形式としては，定数，変数のほか，一般に式を書くことも許されている。

（例） A(3) A(I) A(I+1)

```
110 LET W=0
120 FŌR I=1 TŌ N                    } Σ(x_i−μ)^2
130     LET W=W+(X(I)−M)^2           } の計算
140 NEXT I
150     LET S=SQR(W/N)                 標準偏差σ
160 PRINT "HEIKINTI",M
170 PRINT "HYŌUZYUNHENSA",S
180 END
```

〔備考〕

(1) 後述のDATA文，READ文，RESTORE文を用いれば配列を使わずに$\mu$，$\sigma$を計算することができる。67ページ参照。

(2) $\sigma=\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2-\mu^2}$ という式は誤差が入るので使わない方がよい。

---

† ただし，上限が10以下ならば配列宣言を省略してもよい。

†† 2字以上を許す機種もある。

### 2.4.2　九　九　の　表

掛け算の九九の表を作ってみよう（下の出力例参照）。

**〔設計〕**　いろいろな方法が考えられるが，ここでは，まず，記憶装置の中に九九の表を作り，できあがったら，それを印刷する，というやりかたで作ることにする。

**〔製作〕**　2次元配列と2重のループを用いて次のように書く。

```
10  DIM  A(9,9)
```

Aが9行9列の配列であることを示す。

```
20  FŌR  I=1  TŌ  9
30     FŌR  J=1  TŌ  9
40        LET  A(I,J)=I*J
50     NEXT  J
60  NEXT  I
```

$i$を1から9まで変え，その中で$j$を1から9まで変えて，積$ij$を計算する。

これで配列Aの中に九九の表ができた。

```
70  FŌR  I=1  TŌ  9
80     FŌR  J=1  TŌ  9
90        PRINT  TAB(4*J);A(I,J);
100    NEXT  J
110    PRINT
120 NEXT  I
```

110 の PRINT：これは「改行せよ」という意味になる。

以上6行はBASICで2次元配列を印刷するときの定石。

```
140  END
```

**〔出力例〕**

```
1   2   3   4   5   6   7   8   9
2   4   6   8   10  12  14  16  18
3   6   9   12  15  18  21  24  27
4   8   12  16  20  24  28  32  36
5   10  15  20  25  30  35  40  45
6   12  18  24  30  36  42  48  54
7   14  21  28  35  42  49  56  63
8   16  24  32  40  48  56  64  72
9   18  27  36  45  54  63  72  81
```

**BASICの文法—(6)— 2次元配列その他**

◆　2次元配列の宣言は次のように書く。

　DIM　配列名(第1添字の上限，第2添字の上限)

◆　2次元配列の要素は次のように表す。

　配列名(第1添字，第2添字)

◆　基本BASICでは，配列は2次元までである。3次元以上を許す機種もあるが，一方，簡略BASICには1次元配列しか使えないものがある。

◆　FOR文を入れ子の形で使用することができる（多重の入れ子も可）。

◆　PRINT文の印字項目の所にTAB($n$)と書くと「次の項目は左端から数えて第$n$字目から印字せよ」という意味になる。ただし現在の位置より左にもどることはできない。

◆　拡張BASICでは，

　PRINT　USING"書式"；印字項目の列

という形で書式を細かく指定することができる。書式は，たとえば「最初に空白を2字分入れ，次に符号，次に整数部2桁，それから小数点を打ち，続けて小数部を5桁出力せよ」というのであれば

　"␣␣+##.#####"

というぐあいに書く。

**(注意)**　TABを使わないと左記の例の出力は次のようになる。

```
1  2  3  4  5  6  7  8  9
2  4  6  8  10  12  14  16  18
3  6  9  12  15  18  21  24  27
4  8  12  16  20  24  28  32  36
5  10  15  20  25  30  35  40  45
6  12  18  24  30  36  42  48  54
7  14  21  28  35  42  49  56  63
8  16  24  32  40  48  56  64  72
9  18  27  36  45  54  63  72  81
```

### 2.4.3 応 用 問 題

**1 (ベクトルの計算)**

3次元のベクトル $\boldsymbol{a}=(a_1, a_2, a_3)$, $\boldsymbol{b}=(b_1, b_2, b_3)$ の各成分を読み込み，次の計算をして出力するプログラムを作れ。

i) ベクトルの和

$$\boldsymbol{a}+\boldsymbol{b}=(a_1+b_1,\ a_2+b_2,\ a_3+b_3)$$

ii) ベクトルの差

$$\boldsymbol{a}-\boldsymbol{b}=(a_1-b_1,\ a_2-b_2,\ a_3-b_3)$$

iii) ベクトルの内積 (スカラー積)

$$(\boldsymbol{a},\ \boldsymbol{b})=a_1 b_1+a_2 b_2+a_3 b_3$$

iv) ベクトルの外積 (ベクトル積)

$$\boldsymbol{a}\times\boldsymbol{b}=(a_2 b_3-a_3 b_2,\ a_3 b_1-a_1 b_3,\ a_1 b_2-a_2 b_1)$$

v) ベクトルの長さ

$$|\boldsymbol{a}|=\sqrt{a_1{}^2+a_2{}^2+a_3{}^2}$$

**2 (行列式の値)**

2行2列の行列式の成分 $a_{11}$, $a_{12}$, $a_{21}$, $a_{22}$ を読み込み,

$$\mathrm{D}=\begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix}=a_{11}a_{22}-a_{12}a_{21}$$

の値を計算するプログラムを作れ。また，3行3列の行列式

$$\mathrm{D}=\begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix}$$

についても同様なプログラムを作れ。

**3 (多項式の値)**

次数 $n$, 係数 $a_0$, $a_1$, …, $a_n$ および $x$ の値を読み込み，多項式

$$a_0 x^n+a_1 x^{n-1}+a_2 x^{n-2}+\cdots+a_{n-1}x+a_n$$

の値を計算して印刷するプログラムを作れ。

**《ヒント》** 次のように変形して計算するとよい。

$$((\cdots((a_0 x+a_1)x+a_2)x+\cdots)x+a_{n-1})x+a_n$$

**4 (Fibonacci の数列)**

$a_0 = 0$, $a_1 = 1$,

$$a_{i+1} = a_i + a_{i-1} \qquad (i = 1, 2, \cdots)$$

で定義される数列（Fibonacci の数列という）を $a_{20}$ まで計算して出力するプログラムを作れ。

**5 (2項係数表)**

2項係数（$(a+b)^n$ の展開式の係数。その値は，$n$ 箇のものから $r$ 箇をとる組合せの数に等しい）${}_nC_r$ は，よく知られている公式

$$ {}_nC_r = \frac{n!}{r!(n-r)!}$$

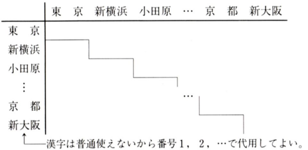
で計算することもできるが，階乗の計算は桁あふれを起こす危険があるから，漸化式

$$ {}_nC_r = {}_{n-1}C_{r-1} + {}_{n-1}C_r \qquad (\text{ただし } {}_nC_0 = {}_nC_n = 1)$$

で計算する方がよい。この漸化式によって2項係数表（$n = 10$まで）を計算して出力するプログラムを作れ。

**6 (駅間距離表)**

新幹線の東京から新大阪までの各駅の，東京からの距離（時刻表に出ている）を読み込み，すべての組合せの2駅間の距離を計算し，下記の形式で出力するプログラムを作れ。




**7** 2次元配列を使用できない機種の場合，1次元配列で代用する方法を工夫せよ。また3次元配列はどのようにして代用すればよいか。

## 2.5 判断，分岐

### 2.5.1 合格，不合格

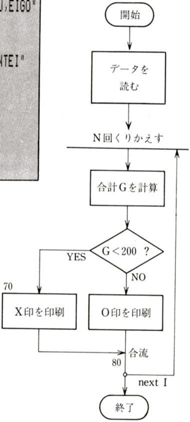
学習塾の入塾試験の判定のため，10人の受験生の

受験番号B　国語の点数K　数学の点数S　英語の点数E

を読み込み，3科目の合計が200点以上ならばO印（合格），200点未満ならばX印（不合格）を出力するプログラムを作ってみよう。

〔プログラム例〕　BASICでは，これを次のように書く。

```
10 DIM B(100),K(100),S(100),E(100)
20 INPUT  N
30 PRINT "BANGOU,KOKUGO,SUUGAKU,EIGO"
40 FOR I=1 TO N
50 INPUT B(I),K(I),S(I),E(I)
60 NEXT I
70 PRINT "BANGOU","TENSUU","HANTEI"
80 FOR I=1 TO N
90 LET G=K(I)+S(I)+E(I)
100 IF G<200 THEN 130
110 PRINT B(I),G,"O"
120 GO TO 140
130 PRINT B(I),G,"X"
140 NEXT I
150 END
```




〔解説〕　行100の文は，「もし G＜200 ならば行130の文に飛び，そうでなければ次の文に進め」という意味である。行120の文は，「無条件に行140に飛べ」という意味で，分岐した処理と重ならないように，分岐した処理の終ったあとで合流させる目的で用いている。

**BASICの文法—（７）—判断，分岐**

◆ IF 文は次の形式で書く。

　　IF　等式または不等式　THEN　行先の行番号

等式または不等式が成立した場合には右記の行番号の文に進む。

等式または不等式が成立しない場合は次の文に進む。

◆ 等式，不等式は次の形式で書く。

　　左辺　等号または不等号　右辺

左辺，右辺には，定数，変数，式などが書ける。

◆ 等号，不等号は次のように書く。

| 数学記号 | ＜ | ≦ | ≧ | ＞ | ＝ | ≠ |
|---|---|---|---|---|---|---|
| BASIC | ＜ | ＜＝ | ＞＝ | ＞ | ＝ | ＜＞ |

◆ GOTO 文は次の形式で書く。

　　GOTO　行先の行番号

これは「無条件に右記の行番号の文に進め」という意味である。

**演習**　1　左記のプログラムを

G≧250 ならばA組（A印）

250＞G≧200 ならばB組（B印）

200＞Gならば不合格（X印）

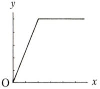
とするように改造せよ。

**（ヒント）**　まずA組に入るかどうかを調べ，次に不合格かどうかを調べればよい。

2　$x$の値を読み，右図のような関数

$$y=\begin{cases}2.5x & (x<2のとき)\\ 5 & (x\geqq 2のとき)\end{cases}$$

の値を出力するプログラムを作れ。




3　整数Nを読み，それが偶数か奇数かを調べて，偶数ならばG，奇数ならばKという文字を出力するプログラムを作れ。

**（ヒント）**　「N／2」と「N／2の整数部」が等しいかどうかを調べればよい（整数部の求めかた→39ページ）。

### 2.5.2　2次方程式の根

2次方程式　$ax^2+bx+c=0$　の係数 $a$，$b$，$c$ を読み込み，根を計算して印刷するプログラムを作ってみよう。

**〔計算式〕**　2次方程式の根の公式

$$x=\frac{-b\pm\sqrt{b^2-4ac}}{2a}$$

を用いるが，コンピューターでは，実数の場合と複素数の場合に分けて扱う必要があるので，まず判別式

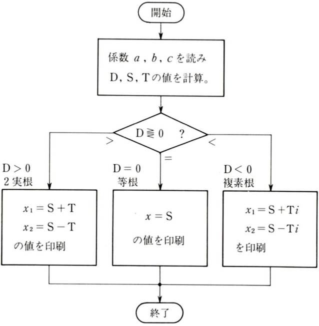
$$D=b^2-4ac$$

の値を計算し，ついでに，あとの便宜を考えて

右辺第1項　$S=\frac{-b}{2a}$　　右辺第2項　$T=\frac{\sqrt{|D|}}{2a}$

を計算しておいて，次のように処理する。

**〔流れ図〕**



〔プログラム例〕

```
10  INPUT A,B,C
20  LET D=B*B-4*A*C
30  LET S=-B/(2*A)
40  LET T=SQR(ABS(D))/(2*A)
50  IF D<0 THEN 140
60  IF D=0 THEN 110
70  REM D>0 NŌ BAAI
80     PRINT "X1=";S+T
900    PRINT "X2=";S-T
100    GŌTŌ 170
110 REM D=0 NŌ BAAI
120    PRINT "X=";S
130    GŌTŌ 170
140 REM D<0 NŌ BAAI
150    PRINT "X1=";S;"+";T;"I"
160    PRINT "X2=";S;"-";T;"I"
170 END
```

**演習** 1 上記のプログラムは $a=0$ の場合を扱うことができない。改造して $a=0$ でも解けるようにせよ。

2 2実根の計算の場合，絶対値の小さい方の根はS±Tの計算の際に桁落ちが起こって精度が悪くなることがあるので，次の式（根と係数の関係）で計算する方がよい。

$$x_{小}=(c/a)/x_{大}$$

ただし $x_{大}$ は「絶対値の大きい方の根」である。上記のプログラムを，このように改造せよ。

### 2.5.3 応用問題

**1 （うるう年）**

西暦年を入力して，その年がうるう年であるかを判定し，その結果を出力するプログラムを作れ。

**《ヒント》** 4で割り切れなければ平年。4で割り切れて，100で割り切れなければうるう年。100で割り切れて，400で割り切れなければ平年。400で割り切れればうるう年。なお，NをMで割り切れるかどうかを調べるには，「N／M」と「N／Mの整数部」が等しいか否かを調べればよい。整数部をとり出すには組み込み関数 INT（39ページ参照）を用いる。

**2 （早生まれ，遅生まれ）**

生年月日（年Y，月M，日Dに分けて入力する）が4月1日以前ならばHAYAUMARE，4月2日以後ならばOSOUMAREと出力するプログラムを作れ。

**3 （大口割り引き）**

ある店のカセットテープの値段は次のようになっている。

バラ（1本単位）だと，330円（1本につき）

1箱（10本入り）は，3000円（1箱につき）

$n$本の値段を計算するプログラムを作れ。

**4 （妥当性のチェック）**

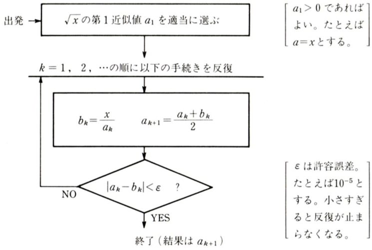
月，日を入力したとき，その値が「あり得ない日付け」（たとえば2月30日とか，13月7日とか，0月－1日など）でないかどうかを調べるプログラムを作れ。

**5 （照合）**

今年来たお年玉つき年賀はがきの番号が配列A（添字1からNまで）に格納されている。今年の当たり番号を入力して，当たりのはがきがあるか無いかを調べるプログラムを作れ。

**6 （収束判定）**

$\sqrt{x}$ の値を求めるには，組み込み関数を使って SQR（X）と書いてもよいが，自分でプログラムを作るのであれば次のようにするとよい。




このようなプログラムを作れ。

**7 （ループ）**

IF 文を用いれば，FOR 文を使わなくても，くりかえしのプログラムを書くことができる。まず，最も簡単な場合として，

```
FŌR I=1 TŌ N
  反復手続き
NEXT I
```

を IF で書いてみよう。次に，一般的な場合

```
FŌR X=A TŌ B STEP C
```

を IF 文で書く方法を工夫せよ（特に，増分Cが負の場合について，注意深く考察せよ）。また，そのやりかたを用いて，$x=0$ から $x=1$ まで，0.1おきの，$e^x$ の数表を作るプログラムを書き，コンピューターで実行させて，結果を検討せよ。

## 2.6 その他の機能

以上で基本的な機能は一通り説明したが，そのほかに「知っておけば便利な機能」がいくつかあるので，本節ではそれらをまとめて説明しておく。はじめて学習するときは，ざっと目を通して先に進み，あとで必要になったときに詳しく読めばよい。なお，簡略BASICの中には本節で紹介する機能を省いてあるものがあるので注意されたい。

### 2.6.1 番号による分岐

IF 文に似た文で，ON 文というのがある。これは

$$\bar{\text{O}}\text{N} \quad n \quad \text{G}\bar{\text{O}}\text{T}\bar{\text{O}} \quad l_1,\ l_2,\ \cdots,\ l_m$$

の形で書き，

$n=1$ ならば行番号 $l_1$ の文に進め

$n=2$ ならば行番号 $l_2$ の文に進め

…

$n=m$ ならば行番号 $l_m$ の文に進め

という意味になる。

〔**使用例**〕　三角形の面積を計算する公式はいろいろある。そこで，まず公式番号 $n$ を読み，

$n=1$ ならば，2辺の長さ $a$，$b$ とそのなす角 $\theta$ より

$$\text{S}=ab\sin\theta/2$$

$n=2$ ならば，底辺の長さ $a$ と高さ $h$ より

$$\text{S}=ah/2$$

$n=3$ ならば，3辺の長さ $a$，$b$，$c$ より，Heron の公式

$$\text{S}=\sqrt{s(s-a)(s-b)(s-c)} \qquad s=(a+b+c)/2$$

で面積Sを計算するプログラムを作ってみよう。$\theta$ を変数名T，$s$ を変数名Zで表し，入力する角度の単位を度とすると，プログラムは次のようになる。

**BASICの文法—(8)—ON文**

◆ ON文は

ŌN $n$ GŌTŌ $l_1$, $l_2$, …, $l_m$

の形で書く。意味は左記の解説のとおり。

◆ $n$の所に式を書いてもよい。

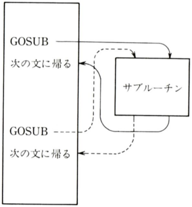
（例） ŌN I+J GŌTŌ 100,200,300

◆ $n$の値は「1以上，$m$以下の整数」にするのが自然であり，安全，確実である。上記以外の場合についての細かい規定もあるが，機種によって扱いかたが違うことがあるので，なるべく上記の形で用いる方がよい。

```
 10 PRINT "N=",
 20 INPUT N
 30 IF N>3 GŌTŌ 10
 40 ŌN N GŌTŌ 50,90,130
 50 PRINT "A,B,THETA=";
 60 INPUT A,B,T
 70 LET S=A*B*SIN(T/57.2958)/2
 80 GŌTŌ 170
 90 PRINT "A,H=";
100 INPUT A,H
110 LET S=A*H/2
120 GŌTŌ 170
130 PRINT "A,B,C=";
140 INPUT A,B,C
150 LET Z=(A+B+C)/2
160 LET S=SQR(Z*(Z-A)*(Z-B)*(Z-C))
170 PRINT "S=";S
180 END
```

### 2.6.2　サブルーチン

プログラムの中のいくつもの箇所で同じ形の処理をしたいことがある。その場合，同じ形のプログラムを何箇所にも書くのはむだであるから，共通の部分をサブルーチン（subroutine，いわば下請けプログラム）という形式で作っておいて，必要な箇所から呼び出して使うと便利である。




サブルーチンを呼び出すには，次の形で書く。

GŌSUB　行番号

右記の行番号は，サブルーチンの入口（最初の文）の行番号である。他のプログラム言語（FORTRANなど）では，サブルーチンにデータを引き渡すために引数というものを書くことができるが，基本BASICには引数がない。そのかわり，変数名が全部共通になっている。

サブルーチンの入口の書きかたに関しては，べつに規則はない（「ここがサブルーチンの始まりである」というような文は不要）。サブルーチンの出口にはRETURNと書く。これは「もとのプログラムに帰れ」という意味で，もとのプログラムの呼び出した所（GOSUB）の次の文に飛んで，そこから続きを実行する。

〔使用例〕　$n$と$r$の値を読み，2項係数（組合せの数）

$$ {}_nC_r=\binom{n}{r}=\frac{n!}{r!(n-r)!} $$

の値を計算して出力するプログラムを作ってみよう。階乗の計算が3箇所にあるので，階乗を文200から始まるサブルーチンで計算することにする。

**BASIC の文法 —（9）— GOSUB 文 その他**

◆ GOSUB 文は次の形式で書く。

　　GŌSUB　サブルーチンの最初の行の行番号

◆ サブルーチンの出口には RETURN 文

　　RETURN

を書く。これを実行すると，呼び出した GOSUB 文の次の文にもどる。

◆ サブルーチンの中で，さらにまたサブルーチンを呼び出す，というように，GOSUB-RETURN を多重に使用できる。

◆ STOP 文は STŌP と書く。「(一時)停止せよ」の意味。

```
 10 INPUT N,R          n と r を読む。
 20 LET K=N          } n! を計算し，結果を
 30 GŌSUB 200        } F1 に入れる。
 40 LET F1=F         }
 50 LET K=R          } r! を計算し，結果を
 60 GŌSUB 200        } F2 に入れる。
 70 LET F2=F         }
 80 LET K=N-R        } (n-r)! の計算。結果
 90 GŌSUB 200        } は F に入る。
100 LET C=F1/(F2*F)
110 PRINT "C=";C
120 STŌP
200 LET F=1            ここからサブルーチン。
210 FŌR I=1 TŌ K      k! の値を計算して結果
220 LET F=F*I          を F に入れる。
230 NEXT I
240 RETURN             サブルーチンの出口。
250 END                全体の終り。
```

### 2.6.3 利用者が定義する関数

組み込み関数（2.2.2 節参照）以外の関数は，自分で作らなければならない。それを一つのプログラムの中の何箇所もで使うのであれば，サブルーチンの形で扱うのがよいであろう。しかし，特に，関数の計算式が1行で書ける場合には，「利用者が定義する関数」という形で扱うと，

引数を設けることができる

算術式の中に書ける

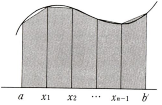
という利点がある。

関数を定義するには DEF 文を用いる（右のページの文法規則参照）。DEF は define の意味である。

〔**使用例**〕 台形公式によって

$$S=\int_a^b f(x)\,dx$$

の値を近似計算するプログラムを作ってみよう。台形公式というのは，右図のように定積分の値を台形の面積の和で近似する公式で，区間 $a$，$b$ を $n$ 等分する場合の計算式は次のようになる。




$$S=\left(\frac{f(a)}{2}+\sum_{i=1}^{n-1}f(x_i)+\frac{f(b)}{2}\right)h$$

ただし，$x_i=a+ih,\quad h=(b-a)/n$

$f(x)$ の計算が3箇所で必要になるので，DEF 文を用いるとよい。以下に示すプログラム例は，$f(x)=1/(1+x^2)$ の場合であるが，DEF 文を取りかえれば他の関数の計算ができる。

```
10  REM  EXAMPLE-2.6.3
20  DEF  FNC(X)=1/(1+X*X)
30  INPUT  A,B,N
40  LET  H=(B-A)/N
50  LET  W=0
60  FŌR  I=1  TŌ  N-1
```

**BASICの文法 —（10）— DEF文**

◆ 関数を定義するには DEF 文を用いる。これは次の形式で書く。

　DEF　関数名（仮引数）＝その関数の計算式

◆ 関数名は 3 文字で，

　先頭 2 文字は FN に限る

　第 3 字は英字に限る

（正しい例）　FNA　FNC

（誤りの例）　TAN　F　FUNCTN

◆ 仮引数が不要ならば無くてもよい。その場合の形は

　DEF　関数名＝その関数の定義式

となる。

◆ 仮引数を書くなら 1 箇に限る（二つ以上は書けない）。

◆ 仮引数としては変数名を書く。本文の変数名と重複可。

◆ 関数は，関数名（実引数）の形で代入文などの式の中に書くことができる。実引数としては，定数，変数のほか，式を書いてもよい。引数なしの関数は関数名だけで引用される。

```
 70    LET X=A+I*H
 80    LET W=W+FNC(X)
 90  NEXT I
100  LET W=FNC(A)/2+W+FNC(B)/2
110  LET S=W*H
120  PRINT "S=";S
130  END
```

**演習**　1　次の関数を DEF 文の形で書け。

(1) $\sinh x = \dfrac{e^x - e^{-x}}{2}$　　(2) $\cosh x = \dfrac{e^x + e^{-x}}{2}$

(3) $\tanh x = \sinh x / \cosh x$

(4) $\arcsin x = \arctan \dfrac{x}{\sqrt{1-x^2}}$

(5) $\arccos x = \pi/2 - \arcsin x$

### 2.6.4 同じデータを何度も使う方法

これまでのプログラム例では，データの入力方法として，

入力の指示を INPUT 文で書く

実際のデータは実行時点に入れる

という方法をとってきたが，もう一つの方法として

データをプログラムと一緒に入れておく

その入力指示を READ 文で書く

ということができる。すなわち，入力すべきデータを

ＤＡＴＡ　１番目のデータ，２番目のデータ，…

（これで書ききれなければ，同じ形式で何枚でも並べる）

の形でプログラムの中（普通はプログラムの最後の所）に書き，それを READ 文で読む。READ 文の書きかたは INPUT 文と同じ形

ＲＥＡＤ　変数名，変数名，…

である。INPUT 文と READ 文の違いは

INPUT 文は，実行時に端末機からデータを読む

READ 文は，DATA 文で書いたデータを読む

という点にある。DATA 文と READ 文の組合せによる方法には次のような利点がある。

1) 統計の計算などの場合，同じデータをいろいろな手法で解析したいことがある。そういうとき，同じデータを何度も入力しないで済む。

2) デバッグのとき，テスト・データを何度も入れるのはめんどうであるが，この方式なら一度入れるだけでよい。

3) この方式だと，RESTORE 文を用いて「いまのデータの最初の位置にもどれ」という指示ができる。こうして，一つのプログラムの中で，同じデータを何回も読むことができる。

要するに，INPUT 文で入力したデータは，すぐ消えてしまってあとに残らないのに対し，DATA 文で書いたデータはプログラムの一部として記憶装置に残っているので，同じデータを何度も利用することができるわけである。

**BASICの文法―(11)―DATA文とREAD文**

◆ DATA 文は次の形式で書く。

DATA データの列

ただし,「データの列」は,データ(すなわち定数)をコンマで区切って並べたものである。

◆ READ 文は次の形式で書く。

READ 変数名の列

ただし,「変数名の列」は,変数名(添字付きでもよい)をコンマで区切って並べたものである。

◆ RESTORE 文は次の形式で書く。

RESTŌRE

◆ 文の意味は左記の解説のとおり。

◆ DATA 文はプログラム中のどこに書いてもよい。

〔使用例〕 2.4.1 節と同じ計算を次のように処理することができる。

```
 10 READ N
 20 LET W=0
 30 FŌR I=1 TŌ N            ⎫
 40    READ X               ⎬ 読みながら合計を
 50    LET W=W+X            ⎪ 計算してゆく。
 60 NEXT I                  ⎭
 70 LET M=W/N
 80 RESTŌRE                   最初のデータにもどる。
 90 LET W=0
100 FŌR I=1 TŌ N            ⎫ 読みながら,「平均
110    READ X               ⎬ 値との差の2乗」
120    LET W=W+(X−M)^2      ⎪ の和を作ってゆく。
130 NEXT I                  ⎭
140 PRINT M,SQR(W/N)
150 END
160 DATA 153                  以下同様にデー
 ⋮                            タを列記する。
```

### 2.6.5 文字列変数

コンピューターの中で文字データを扱いたい場合がよくある（たとえば下記の使用例参照）。文字データの扱いかたは数値データと全く異なるので，数値データを扱うための変数（これまでに述べた普通の変数）とは別に，文字列変数というものがある。その文法の要点を右に記す。基本 BASIC における文字処理機能はあまり強力でないので，本格的な文字処理（たとえば文章の編集，情報検索など）には向かないが，日常の技術計算に必要な機能としては十分で，かなり便利に使えるから，いろいろな使いかたを工夫して活用するとよい。

**〔使用例〕** いくつかの商品の，品名，単価，数量（売上げ数量）を入力し，各品目について（単価）×（数量）=（金額）の計算をして

品名　単価　数量　金額

の形式で作表し，最後に合計金額を出力するプログラムを作ってみよう。品名は文字列変数H¥で表し，単価，数量，金額はT，S，Kで表すことにする。品名欄が＊印ならば「データ終り」と解釈し，合計を出力することにする。作表の関係上，入力はREAD文で行なう（INPUT文で入力すると入力と出力が一緒になってうまくない）。

```
 10 LET G=0
 20 READ H¥,T,S
 30 IF H¥="*" THEN 80            文字列の比較
 40 LET K=T*S
 50 PRINT H¥,T,S,TAB(50);K
 60 LET G=G+K                    金額の印字位置を合計
 70 GOTO 20                      欄と合わせるため。
 80 FOR I=1 TO 11
 90   PRINT "_______"            横けいを引く。
100 NEXT I
```

**BASICの文法—(12)—文字列変数**

◆ 文字列変数の変数名は2字で，1字目は英字，2字目は¥（機種によっては$）とする。（例） S¥ A¥ Y¥

◆ 一つの変数で扱うことのできる文字列の長さは機種によって異なるが，普通の単語や名前を扱うには十分な長さであると考えてよい（標準規格では最大18字としている）。

◆ 文字列変数は，入出力のほか，代入および比較ができる。その場合，文字列定数（すなわち文字列を2重引用符で囲んだもの）を一緒に使うことができる。

◆ 文字列を切ったり，つないだり，中の1字を取り出したり挿入したりする機能や，文字列変数の配列は，標準規格の基本BASICには含まれていない。

```
110 PRINT "GŌUKEI",TAB(50);G
120 STŌP
130 DATA "RINGŌ",200,3
140 DATA "MIKAN",70,10
150 DATA "*",0,0
160 END
```

（130〜150：データの例）

〔出力例〕

```
RUN
RINGO         200       3        600
MIKAN         70        10       700
KAKI          150       5        750
------------------------------------
GOUKEI                           2050
```

### 2.6.6　応　用　問　題

**1　（HAPPY BIRTHDAY）**

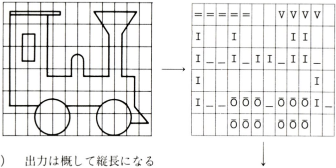
名前を入力し，それを HAPPY BIRTHDAY の歌に組み込んで，

```
HAPPY  BIRTHDAY  TŌ  YŌU.
HAPPY  BIRTHDAY  TŌ  YŌU.
HAPPY  BIRTHDAY  DEAR  名前  SAN.
HAPPY  BIRTHDAY  TŌ  YŌU.
```

と出力するプログラムを作れ。

**2　（まんが）**

コンピューターで絵を描く最も簡単な方法は，まず方眼紙に原画を描き，文字列になおし，文字列を1行分ずつ読み込んでそのまま出力するプログラムに入れてやればよい。



（注意）　出力は概して縦長になるから，上の図のように，方眼紙に縦長の格子を書いて用いるとよい。

データは INPUT 文で入れると出力と一緒になって絵にならないから，DATA 文で書いて READ 文で入力するとよい。

〔出力例〕

```
=====   VVVV
I   I     II
I__I_II_II_
I         I
I__OOO_OOOI_
   OOO OOO
```

**3 (2直線のなす角)**

ベクトル$\boldsymbol{a}$, $\boldsymbol{b}$のなす角$\theta$は

$$\theta=\arccos\frac{(\boldsymbol{a},\boldsymbol{b})}{|\boldsymbol{a}|\cdot|\boldsymbol{b}|}$$

によって求められる。内積$(\boldsymbol{a},\boldsymbol{b})$および絶対値$|\boldsymbol{a}|$, $|\boldsymbol{b}|$のプログラムは既に2.4.3節で作成し，arccosのプログラムは2.6.3節の演習1の(5)で作成してある。これらを組み合わせて，角度$\theta$を計算するプログラムを作れ。データは次の形式で入力するものとする。

| | | |
|---|---|---|
| $n$ | | 次　元 |
| $a_1$ | $b_1$ | 両ベクトルの第1成分 |
| ⋮ | ⋮ | ⋮ |
| $a_n$ | $b_n$ | 両ベクトルの第$n$成分 |

**4 ($\pi$の計算)**

2.6.3節のプログラム例は,

$$S=\int_a^b\frac{dx}{1+x^2}$$

の値を計算するものである。ここで特に$a=0$, $b=1$とすれば

$$S=\int_0^1\frac{dx}{1+x^2}=\Big[\tan x\Big]_0^1=\tan 1=\frac{\pi}{4}$$

の値（近似値）が得られるから，結果Sを4倍すれば円周率$\pi$（の近似値）になるはずである。分割数$n$をいろいろ変えて，どの程度の分割でどの程度の精度が得られるかを調べてみよう。

**5 (入力文字による分岐)**

2.6.1節のプログラムを改造して，公式の選択を番号でなく公式名で行なえるようにせよ。たとえば，入力が

| | | |
|---|---|---|
| FORMULA－1 | ならば | $S=ab\sin\theta/2$ |
| FORMULA－2 | ならば | $S=ah/2$ |
| HERON | ならば | Heronの公式 |

で計算するようにせよ。

## 2.7　操　作　法

BASIC のプログラムを実行するには

　　BASIC システムの起動　　プログラムの入力　　実行

の三つの段階が必要である。

**BASIC システムの起動**　まず最初に

　　電源投入その他，開始時に必要な操作をする。

　　利用登録をする（マイコンの場合は不要）。

　　BASIC システムを呼び出す。

などをしなければならない。その方法は，機種により，また利用形態により異なるから，先生の説明をメモしておこう。

**開始時のきまり文句**

こうして，BASIC システムが動き始めれば，あとは BASIC のコマンドによって操作することができる。

**プログラムの入力**　最初に NEW というコマンドを送り，続けてプログラムを入力する。たとえば

```
NEW                          コマンドには行番号を付けない。
10 INPUT A,B,H
20 LET S=(A+B)*H/2
30 PRINT S
40 END
```

**プログラムの実行**　RUN というコマンドを送ると実行が開始される。いまの例のプログラムだと，最初の文は １０ ＩＮＰＵＴ Ａ，Ｂ，Ｈであるから，まずデータA，B，Hの入力要求(「データを入れて下さい」という合図。普通は？印）が来る。そこでA，B，Hの値を（その順序で）入れる。データの区切りにはコンマを入れる。

```
RUN
? 3.0,8.0,4.5
```

上記のように入力すれば，Aの値は3，Bの値は8，Hの値は4.5になる。次の LET 文を実行するとSの値は24.75になる。その次は ＰＲＩＮＴ Ｓ であるから，Sの値が出力される。

```
24.75
```

**プログラムの修正**　正しい文（の本体）に，修正したい位置に対応する行番号を付けて入力する。

（例）　いまのプログラムを36ページの別解1のように修正するには

```
10 A=3.0
11 B=8.0
12 H=4.5
```

と入力すればよい。上記10のように，既に用いられている行番号と同じものがあれば，以前に入力した文は消されて，新たに入力した文に置き替わる。また上記11のように，既に用いられてる行番号の中間の行番号を入力すれば，入力した文がその位置に挿入される。なお，単に削除したい場合は，行番号だけを入力すればよい。

（例）　36ページの別解2のように修正するには

```
20
30 PRINT (A+B)*H/2
```

と入力すればよい。

〔**プログラムの清書**〕　LIST というコマンドを送ると，その時点におけるプログラム・リストが出力される。

**再実行**　修正したプログラムを実行するにはRUNコマンドを送ればよい。修正しないで，同じプログラムを何回も（いろいろなデータに対して）実行する場合もRUNコマンドによる。

**保存手続き**　こうして作られたプログラムは，普通，一時ファイル（一時的に使用するための記憶場所）に置かれているので，次の人と交代すると（あるいは自分が別のプログラムを作成するためにNEWコマンドを入力すると）消されてしまう。もし次回までとっておきたければ，保存ファイル（情報を長期間保存しておくための記憶場所）に移すか，カセット・テープまたはフロッピー・ディスクに書き込んでおく必要がある。操作法は機種によって異なるから，先生に伺って（ついでに復元の方法，削除の方法なども）メモしておこう。

**待避と復元の方法**

**終了手続き**　終了時には，「BASICの処理をこれで終了する」という手続きと「私の仕事はこれで終ります」という手続きをするのが普通である。

**操作に関するメモ，注意事項**

# 3 プログラミング技法

## 3.1 数値の扱い

**整数の計算**

◆ **関数INT**　整数の計算のコツはINTという関数を活用することである。これは（39ページの表にあるように）

INT（$x$）=「$x$を超えない最大の整数」

という関数である。

（例）　INT（3.18）=3　INT（－3.18）=－4

◆ **余り**　正の整数Mを正の整数Nで割った余りRを求めるには，次のようにすればよい。

```
LET R=M-INT(M/N)*N
```

（例）　M=20，N=6ならば，M／N=3.333…

INT（M／N）=3　R=20－3×6=2

◆ **約数**　正の整数Mが正の整数Nで割り切れるかどうかを調べるには　IF INT(M/N)*N=M THEN …

とすればよい。THENの行先が「割り切れた場合」，分岐せずにIF文の次に進むのが「割り切れなかった場合」である。

◆ **整数計算の誤差**　整数の計算では，普通は誤差は入らないが，BASICでは整数値も浮動小数点表現で扱っているから，オーバーフロー（overflow，あふれ）は起こさないかわり，仮数部桁数を超えれば誤差が入るので注意を要する。なお，標準規格では「有効桁数6桁以上」となっている。普通の計算は6桁でもよいが，金銭の計算などの際には十分に注意されたい。

### 最大公約数と最小公倍数

◆　正の整数M，Nの最大公約数Gを求めるには「両者をM（または N）以下のすべての整数で割ってみる」という方法が最も簡単であるが，もっと能率のよいやりかたとして **Euclid の互除法**というのがある。その手順は次のとおり。

1）　MとNを比較し，$M<N$ならば交換して$M \geqq N$にする。

2）　MをNで割った余りRを求める。

3）　余りが0かどうかを調べて，

- 0ならば計算終了（最大公約数はNである）
- 0でなければ，新$M=N$，新$N=R$として2）にもどる。

（例）　$M=20$，$N=12$の場合，

20÷12の余りは8　　新$M=12$，新$N=8$

12÷8の余りは4　　新$M=8$，新$N=4$

8÷4の余りは0だから計算終了。最大公約数は4。

◆　M，Nの最小公倍数Lを求めるには，MとNを掛けて，MとNの最大公約数Gで割ればよい。すなわち，$L=MN/G$。

〔プログラム例〕

```
 10 INPUT M,N
 20 IF M>=N THEN 60
 30 LET W=M          ┐
 40 LET M=N          ├ MとNを変換
 50 LET W=W          ┘
 60 LET R=M-INT(M/N)*N     余りの計算
 70 IF R=0 THEN 110
 80 LET M=N
 90 LET N=R
100 GOTO 60
110 PRINT "GCD=";N
120 END
```

**分数の計算**

分数は小数になおして計算するのが最も簡単であるが，分数の形のまま扱うことも可能である。それには，筆算の場合と同様，分母の計算，分子の計算，結果の約分などを，プログラムで書いてやればよい。

**〔プログラム例〕** 分数の加算のプログラムを作ってみよう。公式は

$$\frac{B}{A}+\frac{D}{C}=\frac{BC+AD}{AC}$$

であるから，最も簡単に済ませるつもりなら，次のようにすればよい。

```
10 INPUT A,B,C,D
20 LET E=A*C
30 LET F=B*C−A*D
40 PRINT B;"/";A;"+";D;"/";
  (続き)  C;"=";F;"/";E
50 END
```

もっとていねいにやるとすれば，次のようになる。

1） AとCの最小公倍数Lを求める（前ページ参照）
2） E＝L
3） F＝B＊（L／A）＋D＊（L／C）
4） EとFの最大公約数Gを求める（前ページ参照）
5） 新E＝E／G，新F＝F／G
6） 必要ならば，過分数を帯分数になおす

**複素数の計算**

これも同様に，実数部と虚数部それぞれの計算をすればよい。公式は

$$(a+ib)\pm(c+id)=a\pm c+i(b\pm d)$$

$$(a+ib)\cdot(c+id)=(ac-bd)+i(ad+bc)$$

$$\frac{a+ib}{c+id}=\frac{ac+bd}{c^2+d^2}+i\frac{bc-ad}{c^2+d^2}$$

$$|a+ib|=\sqrt{a^2+b^2} \qquad \overline{a+ib}=a-ib$$

**時分秒の計算**

時，分，秒の単位を併用して時刻を表すことは日常よく行なわれている。その加減算を行なうには

- 60進法の加減算のプログラムを作る
- すべてを秒単位に換算して加減算を行なう

の二つの方法がある。IF 文の練習や$n$進法演算の扱いに慣れるためには前者もよいが，実務には後者の方が簡単でよい。

〔**プログラム例**〕「$h_1$ 時 $m_1$ 分 $s_1$ 秒から $h_2$ 時 $m_2$ 分 $s_2$ 秒までは何時間何分何秒か？」という計算（時分秒の引き算）をするプログラムを作ってみよう。

```
 10 PRINT "H1,M1,S1=";              } データの入力
 20 INPUT  H1,M1,S1
 30 PRINT "H2,M2,S2=";
 40 INPUT  H2,M2,S2
 50 LET T1=3600*H1+60*M1+S1         } 秒に
 60 LET T2=3600*H2+60*M2+S2         } 換算
 70 LET T=T2-T1                       引き算をする
 80 LET H=INT(T/3600)
 90 LET R=T-3600*H                    3600で割った余り
100 LET M=INT(R/60)
110 LET S=R-60*M                      60で割った余りが秒
120 PRINT H;"ZIKAN";
130 PRINT M;"HUN";
140 PRINT S;"BYŌU"
150 END
```

〔**出力例**〕

```
H1,M1,S1=? 3,45,11
H2,M2,S2=? 7,20,30
 3 ZIKAN 35 HUN 19 BYOU
```

**演習**　**1**　フィートとインチで表された長さの加算を行なうプログラムを作れ。

## 丸め

**正の数の，小数点以下を丸める方法**

◆ **切り捨て** 関数INTを用いればよい。

Y＝INT（X）

◆ **切り上げ** 符号を逆にして切り捨てて符号をもとにもどす。

Y＝－INT（－X）

◆ **四捨五入** 0.5を加えて切り捨てる。

Y＝INT（X＋0．5）

**負の数の，小数点以下を丸める方法** 負数の場合の扱いは，

｛絶対値について，切り上げ，切り捨て，四捨五入する
　符号つきの意味で，切り上げ，切り捨て，四捨五入する

の二通りある。前者ならば，次のようにする。

◆ **切り捨て** Y＝－INT（－X）

◆ **切り上げ** Y＝INT（X）

◆ **四捨五入** Y＝－INT（－X＋0．5）

後者ならば，次のようにする。

◆ **切り捨て** Y＝INT（X）

◆ **切り上げ** Y＝－INT（－X）

◆ **四捨五入** Y＝INT（X＋0．5）

**小数点以外の所で丸める方法**

適当な係数$10^n$を掛けて「小数点の所で丸める」という形の問題に変換し上記の処理をしてから$10^n$で割ってもとにもどす。

（例） 小数点以下第3位を四捨五入するには

Y＝INT（100＊X＋0．5）／100

ただし，2進法や16進法のコンピューターでは，10進法の小数を正確に表現することができないので厳密な意味での切り捨て，切り上げ，四捨五入は不可能である。

（例） 0.15は2進法で正確に表現できない。その近似値が0.15より小さければ，四捨五入して0.1になりうる。

**【3.1 節の応用問題】**

**1　（素数）**

正の整数Nを入力し，Nが素数であるか否かを判定するプログラムを作れ。

**《ヒント》**　2から $\sqrt{N}$ までのすべての整数で割ってみればよい。NがMで割り切れるかどうかを調べるには，

N／M−INT（N／M）

が0かどうかを調べればよい。

**2　（素数表）**

1000までの素数を全部求めて出力するプログラムを作れ。

**《ヒント》**　「Eratosthenesのふるい」という方法を用いるとよい。1000までの素数を求めるのであれば，まず2から1000までの表を用意し，2の倍数全部，3の倍数全部，…に印をつける（ただし，1倍，すなわち自分自身は除く）。こうして $\sqrt{1000}$ 以下のすべての数の倍数に印をつけたとき，残ったものが素数である。

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 2 | 3 | ~~4~~ | 5 | ~~6~~ | 7 | ~~8~~ | ~~9~~ | ~~10~~ | … |

これをコンピューターで行なうには，寸法1000の配列Aを用意し，その要素全部に0を入れておき，

```
100 FŌR K=2 TŌ SQR(1000)
110   FŌR I=2 TŌ 1000/K
120     LET L=I*K
130     IF L>1000 THEN 160
140     LET A(L)=1
150   NEXT I
160 NEXT K
```

というぐあいに処理していけばよい。なお，行100と行110の間で，Kが素数であるかどうかを調べ，素数でなければ行160までスキップするようにすれば，むだが少なくなり，処理時間を大幅に短縮できる。また，行110から始まるループも，よく考えてみると，K＝I－1までは既に

印がついているのであるから，

```
FŌR I=K TŌ 1000/K
```

としてよい。

**3 （素因数分解）**

正の整数Nを入力し，これを素数の積の形に分解するプログラムを作れ。ただしN≦10000とし，22までの素数表を記憶させて利用してよい。

**（ヒント）** 下記の要領で処理すればよい。

K＝2，3，5，7，11，13，…の順に

K＞Nならば終了。

→NをKで割ってみる。割り切れなければ—

—割り切れたならば—

Kを出力。

新N＝現N／K

N＞1ならば前にもどって反復

N＝1ならば終了。

次のKへ←

100までの素数は下記の25箇である。

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2 | 3 | 5 | 7 | 11 | 13 | 17 | 19 | 23 | 29 | 31 | 37 | 41 |
| 43 | 47 | 53 | 59 | 61 | 67 | 71 | 73 | 79 | 83 | 89 | 97 | |

出力は，たとえばN＝24の場合

2　2　2　3

という形にするのが最も簡単であるが，工夫すれば

24＝2＊2＊2＊3

（またはこれに多少，空白が入った形）にすることが可能であろう。

**4 （分数の加減乗除）**

二つの分数B／A，D／Cを入力し，その和，差，積，商を分数の形で計算して出力するプログラムを作れ。

**（ヒント）** 77ページのプログラムを参考にするとよい。

**5 (複素数の加減乗除)**

二つの複素数 $z_1=a+ib$, $z_2=c+id$ を読み込み，その和，差，積，商を計算して出力するプログラムを作れ。

《ヒント》 77ページの公式を参考にするとよい。

**6 (年月日の引き算)**

西暦 $y_1$ 年 $m_1$ 月 $d_1$ 日から，西暦 $y_2$ 年 $m_2$ 月 $d_2$ 日までの日数を計算するプログラムを作れ。

《ヒント》 年，月，日を別個に引き算して，引けなければ上位の値から借りてくる，という方法も考えられるが，月の日数が一定でないため，かなり複雑なプログラムになる。それよりも，一定の基点（たとえば，1600年1月0日）からの日数を計算するプログラムを作り，その日数で引き算を行なう方が簡単である。

まず，1月0日から $m$ 月 $d$ 日までの日数を計算する方法を考えてみるとよい。それには，各月の0日が1月0日から何日目になるかを電卓で計算して表を作り記憶させておくと簡単である。

| 1月 | 2月 | 3月 | 4月 | 5月 | 6月 |
|---|---|---|---|---|---|
| 0 | 31 | 59 | 90 | 120 | 151 |
| 7月 | 8月 | 9月 | 10月 | 11月 | 12月 |
| 181 | 212 | 243 | 273 | 304 | 334 |

これは平年の場合である。うるう年には，3月以後，上記の値に1を加える。上記の値に $d$ を加えれば，1月0日から $m$ 月 $d$ 日までの日数N1がわかる。

次に，1600年1月0日から，$y$ 年1月0日までの日数N2を計算する。それにはいろいろな方法が考えられるが，最も簡単かつ明快なのは，ひとまず1年を365日として計算し，それにうるう年のぶんの補正を加える，という方法である。1年を365日として計算すると，N2の概算値は

$$365\times(y-1600)$$

となる。これに「1600年から$y$年までの，うるう年の回数」を加えればよい。その回数は組み込み関数INTを用いて

```
INT((Y−1600)/4)+1
```

という式で求めることができる。この式は「西暦を4で割り切れる年はうるう年である」という規則に基づくものである†。ところで，暦法には

100で割り切れる年は平年とする

ただし400で割り切れる年はうるう年とする

という規則もある†。この補正を行なうには，

INT((Y−1600)/100)+1を引き，

INT((Y−1600)/400)+1を加える

とすればよい。以上をまとめると，計算式は次のようになる。

```
Z=Y−1600
N2=365*Z+INT(Z/4)−INT(Z/100)
  (続き)                +INT(Z/400)+1
IF Z>0 THEN N2=N2+1
N1=C(M)+D          ただしCは左のページに示した表
うるう年の3月以後はN1に1を加える。
N=N1+N2
```

**7 （曜日）**

西暦$y$年$m$月$d$日が何曜日かを調べるプログラムを作れ。

**（ヒント）** 前問のヒントで説明した「1600年1月0日からの日数」Nを計算し，それを7で割った余りをRとすれば，次のようになる。

| R | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|
| 曜日 | 金 | 土 | 日 | 月 | 火 | 水 | 木 |

---

† これは1582年にGregory 13世が定めた，いわゆるグレゴリオ暦の規則である。1581年以前には適用できない。

## 3.2 データ処理

**数える**

箇数を数えるには，箇数を記録するための変数（カウンター）を用意し，最初そこに0を入れておき，順に1を加えていけばよい。複数箇のカウンターを用いれば，分類して数えることもできる。

**〔例題1〕** 50日間の，正午の気温の観測値 $T_1$，$T_2$，…，$T_{50}$ を読み30℃以上であった日数K，25℃以下であった日数Lを数えて出力するプログラムを作れ。

**〔プログラム例〕** K，Lをカウンターに用いれば次のようになる。

```
10 LET K=0                            } カウンター
20 LET L=0                            } をリセット
30 FŌR I=1 TŌ 50
40    INPUT T                           気温を入力
50    IF T>=30 THEN 80  ─┐
60    IF T<=25 THEN 100 ─┼─┐
70    GŌTŌ 110           │ │
80    LET K=K+1  ←───────┘ │            30℃以上
90    GŌTŌ 110             │
100   LET L=L+1  ←─────────┘            25℃以下
110 NEXT I
120 PRINT "T>=30␣";K;"NITI"           } 結果の
130 PRINT "T<=25␣";L;"NITI"           } 出力
140 END
```

**〔例題2〕** 1000人の学生の都道府県番号（右表参照）$N_1$，$N_2$，…，$N_{1000}$ を読み，都道府県別の人数を数えて出力するプログラムを作れ。

**〔プログラム例〕** 上限47の配列Kをカウンターに用いることにする。

```
10 DIM K(47)
20 FŌR I=1 TŌ 47                      } カウンター
30    LET K(I)=0                      } をリセット
```

```
 40 NEXT I
 50 FŌR I=1 TŌ 1000
 60   INPUT N
 70   IF N>47 THEN 140          (入力の誤り)
 80   LET K(N)=K(N)+1           府県別の計数
 90 NEXT I
100 FŌR I=1 TŌ 47
110   PRINT I,K(I)              結果の出力
120 NEXT I
130 STŌP
140 PRINT "DATA GA ŌKASII"
150 END
```

**演習** 1 200人の学生の生年月日を入力し，生年別，月別，日別の人数を数えて出力するプログラムを作れ。

2 例題1と同じデータを読み，「前日より気温が高くなった日」を数えて出力するプログラムを作れ。

3 都道府県名をローマ字で出力するように例題2を改造せよ。
**（ヒント）** 都道府県名を番号順にDATA文で入れておいて，それを文字列変数にREADしてPRINTするのが一案である。

---

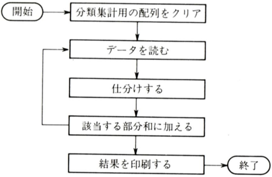
**メモ** **都道府県コード**（JIS C 6020）

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 01 | 北海道 | 11 | 埼玉県 | 21 | 岐阜県 | 31 | 鳥取県 | 41 | 佐賀県 |
| 02 | 青森県 | 12 | 千葉県 | 22 | 静岡県 | 32 | 島根県 | 42 | 長崎県 |
| 03 | 岩手県 | 13 | 東京都 | 23 | 愛知県 | 33 | 岡山県 | 43 | 熊本県 |
| 04 | 宮城県 | 14 | 神奈川県 | 24 | 三重県 | 34 | 広島県 | 44 | 大分県 |
| 05 | 秋田県 | 15 | 新潟県 | 25 | 滋賀県 | 35 | 山口県 | 45 | 宮崎県 |
| 06 | 山形県 | 16 | 富山県 | 26 | 京都府 | 36 | 徳島県 | 46 | 鹿児島県 |
| 07 | 福島県 | 17 | 石川県 | 27 | 大阪府 | 37 | 香川県 | 47 | 沖縄県 |
| 08 | 茨城県 | 18 | 福井県 | 28 | 兵庫県 | 38 | 愛媛県 | | |
| 09 | 栃木県 | 19 | 山梨県 | 29 | 奈良県 | 39 | 高知県 | | |
| 10 | 群馬県 | 20 | 長野県 | 30 | 和歌山県 | 40 | 福岡県 | | |

**合計と分類集計**

◆ 合計を求めるには，部分和を記憶するための変数(いわば累計器)を設け，最初そこを0にしておいて，値を順に加算していけばよい。

◆ 分類集計を行なうには，上記のような「累計器」を分類の数だけ用意し，最初はどれにも0を入れておき，各データを仕分けして該当する「累計器」に加算していけばよい。




**〔例題〕** 家計簿の整理をするプログラムを作れ。データとしては

日付　支出費目コード　金額

を入れるものとする。日付は年月を省き日のみで，もしも日付が0ならば，「入力データの終り」と解釈することにする。支出費目コードは，たとえば，食費が1，住居光熱費が2，というように1から5まであるとする。処理としては，日付別の集計と，支出費目別の集計を行なうことにする。

**〔設計〕** 日付，支出費目コード，金額を変数名D，C，Aで表し，日付別累計に配列K，費目別累計に配列Lを用いることにする。

| 配列K | 日付 | 1 | 2 | 3 | … | … | … | … | 31 |
|---|---|---|---|---|---|---|---|---|---|
| | 小計 | $k_1$ | $k_2$ | $k_3$ | … | … | … | … | $k_{31}$ |

| 配列L | 費目 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| | 小計 | $l_1$ | $l_2$ | $l_3$ | $l_4$ | $l_5$ |

```
10  DIM K(31),L(5)
20  FŌR D=1 TŌ 31              配列Kを
30     LET K(D)=0              クリア
40  NEXT D
50  FŌR C=1 TŌ 5               配列Lを
60     LET L(C)=0              クリア
70  NEXT C
80  INPUT D,C,A
90  IF D=0 THEN 150
100 IF D>31 THEN 250
110 IF C>5 THEN 250
120 LET K(D)=K(D)+A
130 LET L(C)=L(C)+A
140 GŌTŌ 80
150 PRINT "HIZUKE-BETU GŌUKEI"
160 FŌR D=1 TŌ 31              日付別分類
170    PRINT D,K(D)            結果の印刷
180 NEXT D
190 PRINT
200 PRINT "HIMŌKU-BETU GŌUKEI"
210 FŌR C=1 TŌ 5               費目別分類
220    PRINT C,L(C)            結果の印刷
230 NEXT C
240 STŌP
250 PRINT "IRENAŌSITE KUDASAI"
260 GŌTŌ 80                    入力ミスなので入れなおしを要求
270 END
```

**演習** 1 上記のプログラムに見出しの印刷などを加えて使いぐあいのよいプログラムに仕上げよ。

2 日付別，かつ，費目別の分類合計を行なうにはどうすればよいか。

**行和と列和**　データ $x_{ij}(i=1,\ \cdots,\ m,\ j=1,\ \cdots,\ n)$ を読み，その

行和　$S_i=\sum_{j=1}^{n}x_{ij}$

列和　$T_j=\sum_{i=1}^{m}x_{ij}$

総計　$Z=\sum_{i=1}^{m}\sum_{j=1}^{n}x_{ij}$

| $i$ ＼ $j$ | 1 | 2 | … | $n$ | 行和 |
|---|---|---|---|---|---|
| 1 | $x_{11}$ | $x_{12}$ | … | $x_{1n}$ | $S_1$ |
| 2 | $x_{21}$ | $x_{22}$ | … | $x_{2n}$ | $S_2$ |
| ⋮ | | … | | | ⋮ |
| ⋮ | | … | | | ⋮ |
| $m$ | $x_{m1}$ | $x_{m2}$ | … | $x_{mn}$ | $S_m$ |
| 列和 | $T_1$ | $T_2$ | … | $T_n$ | $Z$ |

を計算して出力するプログラムを作ってみよう。このような形の計算は会計事務や統計処理によく現われる。

〔**プログラム例**〕　練習であるから $m$，$n$ は5以下としてよいであろう。

```
 10 DIM X(5,5),S(5),T(5)
 20 PRINT "M,N=";
 30 INPUT M,N
 40 FŌR I=1 TŌ 5
 50   LET S(I)=0
 60   LET T(I)=0
 70 NEXT I
 80 LET Z=0
 90 FŌR I=1 TŌ M
100   FŌR J=1 TŌ N
110     PRINT "X(";I;",";J;")=";
120     INPUT X(I,J)
130     LET S(I)=S(I)+X(I,J)
140     LET T(J)=T(J)+X(I,J)
150     LET Z=Z+X(I,J)
160   NEXT J
170 NEXT I
180 PRINT
190 FŌR I=1 TŌ M
200   FŌR J=1 TŌ N
210     PRINT X(I,J),
```

```
220     NEXT J
230     PRINT "␣I␣";S(I)
240   NEXT I           ↑────── たて線を引くため
250   FŌR J=1  TŌ  N+1
260     PRINT "----------------";
270   NEXT J
280   PRINT "---"
290   FŌR J=1  TŌ  N
300     PRINT  T(J),
310   NEXT J
320   PRINT  "␣I␣";Z
330   END
```

〔出力例〕

```
122.6          351.7          I  474.3
256.8          489            I  745.8
523.6          872.3          I  1395.9
--------------------------------------------
903            1713           I  2616
```

**演習**　**1**　上記のプログラムを改造して

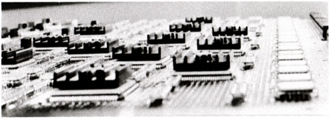
行ごとの平均値　　列ごとの平均値

行ごとの分散　　　列ごとの分散

$S_1$, $S_2$, …, $S_m$ の分散

$T_1$, $T_2$, …, $T_n$ の分散

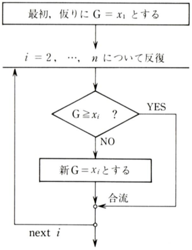
を計算できるようにせよ。このような計算は統計解析の際にしばしば必要になる。



**大型コンピューター（M-200H）の内部**

**最大値，最小値**

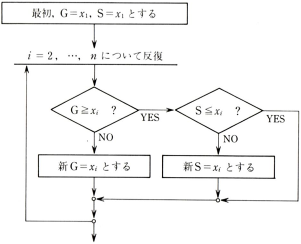
◆　一組の値 $x_1$，$x_2$，…，$x_n$ の最大値Gを求めるには，次のようにすればよい。




（反復終了時のGの値が最大値である）

◆　最小値も同様な要領で求めることができる。

◆　最大値Gと最小値Sを同時に求めるには次のようにする。




（反復終了時のG，Sの値が最大値，最小値である）

〔例題〕 60人の学生の学生番号Kと成績Xを読み，成績の最高点とその学生の学生番号を出力するプログラムを作れ。

〔プログラム例〕 最大値の記憶場所Gのほかに，最大値に対応する番号の記憶場所Lを用いる。

```
10  LET N=60
20  INPUT K,X          ┐ 最初は1人目の学生が
30  LET G=X            │ 仮りの王座につく。
40  LET L=K            ┘
50  FŌR I=1 TŌ N
60    INPUT K,X
70    IF G>=X THEN 100 ──┐
80    LET G=X            │ } 王座交代
90    LET L=K            │
100 NEXT I    ←──────────┘
110 PRINT "SAIKŌUTEN",G
120 PRINT "BANGŌU",L
130 END
```

〔備考〕 このプログラムだと，最高点の同点者がいても1人しか出力されない。同点者を全員出力するには，最高点がわかったあとで，また最初にもどって，全員について同点か否かを調べるのが最も簡単である。

**演習**

1 最小値を求めるプログラムを作れ。

2 最大値と最小値を同時に求めるプログラムを作れ。

3 「絶対値の最大値」を求めるにはどうすればよいか。

4 60人の学生の性別（JIS C 6264 によれば，性別コードは男1，女2である）と成績を読み込み，男女別の最高点を求めて出力するプログラムを作れ。

5 ある中学（3学年，各5組）の水泳記録会の記録を

　　学年　組　個人番号　50m自由形のタイム

の形式で入力し，全校での1位，学年別の1位，クラス別の1位，を調べて，そのタイムと個人番号を出力するプログラムを作れ。

6 「上から2番目の値」を求めるにはどうすればよいか。

### 大きさの順に並べる

一組のデータ $x_1$，$x_2$，…，$x_n$ を，大きさの順（大きい順，または小さい順）に並べかえたいことがよくある。このような処理を**ソート**（**整列**，sorting）という。それには大別して

内部ソート　主記憶装置の中でソートを行なう
外部ソート　外部記憶装置を用いてソートを行なう

がある。後者は，データが大量にあって主記憶装置に入りきれない場合に用いられる。内部ソートの方法としては，最大値法（最小値法），交換法，ヒープ法などがある。外部ソートには，普通，マージ・ソート（merge sort，併合法）という方法が用いられる。

◆　**最大値法**　大きい順に並べるには，まず最大のものをさがし，次にそれを除いた残りの中で最大のものをさがし，以下同様に「残りの中での最大値」をさがして並べてゆけばよい。

```
 10 DIM X(100)
 20 INPUT N                       }
 30 FŌR I=1 TŌ N                  } データの読み込み
 40   INPUT X(I)                  }
 50 NEXT I                        }
 60 FŌR I=1 TŌ N-1                  第 i 回めには x_{i+1}
 70   LET M=I                       から x_n までの間
 80   FŌR J=I+1 TŌ N                の最大値をさがす
 90     IF X(M)>X(J) THEN 110
100     LET M=J
110   NEXT J
120   LET W=X(I)                  } 今回の最大値X(M)をI
130   LET X(I)=X(M)               } 番めの位置にもってくる
140   LET X(M)=W                  } ための入れかえ作業
150 NEXT I
160 FŌR I=1 TŌ N                  }
170   PRINT X(I)                  } 結果の印刷
180 NEXT I                        }
190 END
```

◆ **交換法**　これの基本形は「隣り合う二つのデータを比較し，

正順（望ましい順序）になっていればそのまま

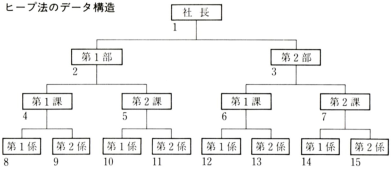
逆順（逆の順序）になっていれば入れ替える

という処理を，列の最初から最後まで何回もくりかえす」という方法である。詳しく検討してみると，反復回数は最大 $n-1$ 回で十分であり，第 $i$ 回めには「列の最初から $n-i$ 番めまで」調べれば十分であることがわかる。

```
 10 DIM X(100)
 20 INPUT N                  ┐
 30 FŌR I=1 TŌ N             │ データの読み込み
 40   INPUT X(I)             │
 50 NEXT I                   ┘
 60 FŌR M=N-1 TŌ 1 STEP -1
 70   FŌR J=1 TŌ M
 80     IF X(J)>X(J+1) THEN 120
 90     LET W=X(J)           ┐
100     LET X(J)=X(J+1)      │ 入れかえ
110     LET X(J+1)=W         ┘
120   NEXT J
130 NEXT M
140 FŌR I=1 TŌ N             ┐
150   PRINT X(I)             │ 結果の印刷
160 NEXT I                   ┘
170 END
```

◆ **ヒープ法**　データの個数が 100 ぐらいまでならば，以上の方法で実用上十分である。しかしデータの量がもっと多くなると，これらの方法は計算の手間が急増して不利になる。その理由は，要するに，反復第 $i$ 回の処理の際に前回（第 $i-1$ 回）までの情報が十分に生かされないためである。これを改めるため，トーナメント試合の方式を取り入れたのがヒープ法である。

**ヒープ法のデータ構造**




**〔原理〕**　上図のような組織の会社を考える。当社では常に

（直系の上司の実力）＞（直系の部下の実力）

という関係が成り立つようにしている。この関係を維持するため，人事異動は次のような方式で行なっている。

1）社長が退任したときは，第1部長と第2部長の実力を比較し，実力のある方を社長に昇格させる。その後任には，その部の第1課長と第2課長の内，実力のある方を昇格させ，その後任には，その課の第1係長と第2係長の内の実力のある方を昇格させる。以下同様。

2）新入社員を採用したときは，一応，最下位に配属するが，ただちにその直属の上司と実力を比較し，逆順なら交代させ，交代した場合はもう一つ上の上司と実力を比較し，逆順ならば交代させる。以下同様。

3）会社設立の際には，社員なしの状態から始めて，上図の番号順にポストをうめてゆき，そのたびに上記2）の処理を行なう。

上記のような組織をヒープ（heap）という。これを用いてソートを行なうには，まずデータ $x_1$，$x_2$，…，$x_n$ を上記3）によってヒープの形に並べ，次に「社長」の所から一人ずつ引き抜いて順に並べ，その後任を上記1）の方法でうめてゆけばよい。

**〔プログラム例〕**　このような処理を BASIC で行なうには上図のような社員番号（配列Xの添字）を用いるとよい。こうすると上司の番号は「部下の番号の 1/2 の整数部」，また部下の番号は「上司の番号の2倍またはそれプラス1」となり容易に上下のデータを得ることができる。

```
10 DIM X(100)
20 INPUT N
30 FOR L=1 TO N
40 INPUT X(L)
50 NEXT L
60 REM ***** PART-1 *****
70 FOR L=2 TO N
80 LET J=L
90 LET I=INT(J/2)
100 IF X(I)>=X(J) THEN 160
110 LET W=X(I)
120 LET X(I)=X(J)
130 LET X(J)=W
140 LET J=I
150 IF J>1 THEN 90
160 NEXT L
170 REM ***** PART-2 *****
180 FOR M=N TO 2 STEP -1
190 PRINT X(1)
200 LET I=1
210 LET J1=2*I
220 LET J2=2*I+1
230 IF J1=M THEN 280
240 IF J1>M THEN 320
250 IF X(J1)>X(J2) THEN 280
260 LET IO=J2
270 GO TO 290
280 LET IO=J1
290 LET X(I)=X(IO)
300 LET I=IO
310 GO TO 210
320 LET X(IO)=X(M)
330 LET I=IO
340 LET K=INT(I/2)
350 IF X(K)>=X(I) THEN 410
360 LET W=X(I)
370 LET X(I)=X(K)
380 LET X(K)=W
390 LET I=K
400 IF I>1 THEN 340
410 NEXT M
420 PRINT X(1)
430 END
```

【3.2節の応用問題】

**1　(素数の分布)**

3.1節の応用問題2（素数表）のプログラムを改造して，

100以下の素数の箇数

101から200までの範囲の素数の箇数

…

901から1000までの範囲の素数の箇数

を数えて出力するプログラムを作れ。

**2　(欠席日数)**

あるクラスで，1年間，毎日，欠席者の生徒番号（1～40の一連番号を用いている）を記録したメモがある。その記録を入力し，

1日も休まなかった生徒

1日だけ休んだ生徒

2日休んだ生徒

…

に分類して，それぞれに該当する生徒番号を出力するプログラムを作れ。

**3　(金種別計算)**

給料の支払いは銀行振込みの会社が多くなったが，まだ現金で支給している所も少なくない。現金で渡す場合には，各人の給料袋に分けて入れる関係上，1万円札のほかに，5千円札，千円札，500円札，百円玉，十円玉を，必要な枚数だけ用意しなければならない。その必要枚数を計算するのが金種別計算である。従業員 $n$ 人（$n \leqq 100$とする）の今月の現金支給額 $a_1$，$a_2$，…，$a_n$ を読み込み，金種別計算を行なうプログラムを作れ。

《ヒント》　各個人の必要枚数は次のようにして計算できる。

（現金支給額）÷（1万）＝（1万円札枚数）余り（1万円未満）

（1万円未演）÷（5千）＝（5千円札枚数）余り（5千円未満）

（5千円未満）÷（　千）＝（　千円札枚数）余り（　千円未満）

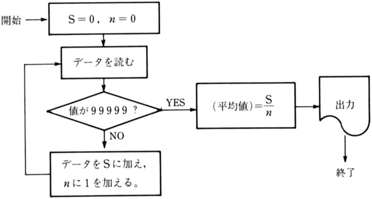
以下同様

**4　(不定箇数の読み込み)**

一組のデータ $a_1$, $a_2$, …, $a_n$ を入力するとき，データの箇数 $n$ がわかっていれば，$n$ を最初に入力することができるが，場合によっては，$n$ がわからないとか，数えるのがめんどうだから数えないで済ませたいということがある。そのような場合には，最後のデータに何か印をつけるか，あるいは最後のデータの次に何か特別な値を入れるかして，入力を終了させればよい。

上記のような方式で，不定箇数のデータを読み込み，その平均値を計算するプログラムを作れ。

**《ヒント》**　たとえば最後のデータの次に 99999 を入れることにして，次のように処理すればよい。




**5　(相関係数)**

データ $x_1$, $x_2$, …, $x_n$ と $y_1$, $y_2$, …, $y_n$ を読み込み，相関係数

$$r=\frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{n\sigma_x\sigma_y}$$

(ただし，$\bar{x}$, $\bar{y}$, $\sigma_x$, $\sigma_y$ は，$x_1$, $x_2$, …, $x_n$ および $y_1$, $y_2$, …, $y_n$ の平均値および標準偏差）を計算するプログラムを作れ。

## 3.3 グラフ表示

グラフ表示の方法には

- 文字出力機能だけを用いる
- グラフ表示機能を用いる

の二通りがある。後者の方がきれいで精密なグラフを描けるが，そのための装置を必要とし，使用法が機種によってそれぞれ違うので，ここでは前者について説明する。

### 3.3.1 棒グラフ

データ $x_1$，$x_2$，…，$x_n$ の値を棒グラフで表示するには，概略，次のようにすればよい。

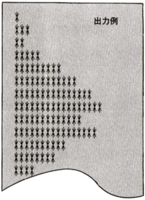
1）1文字でいくつを表すかを決める。その係数を $c$ とする

2）各 $x_i$ に $c$ を掛けて四捨五入し，文字数 $m_i$ を算出する

3）適当な文字を $m$ 回くりかえして（一つの行に）出力する

〔プログラム例〕

```
 10 DIM X(20)
 20 INPUT N,C                 NとCを指定
 30 FŌR I=1 TŌ N            ┐
 40   INPUT X(I)            │ データを読む
 50 NEXT I                  ┘
 60 FŌR I=1 TŌ N
 70   LET M=C*X(I)
 80   FŌR J=1 TŌ M
 90     PRINT "*";
100   NEXT J
110   PRINT
120 NEXT I
130 END
```



### 3.3.2 曲線 $y=f(x)$ のグラフ

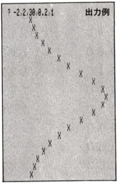
関数 $y=f(x)$ のグラフを $a \leqq x \leqq b$ の範囲で描くには，係数 $c$，原点位置 $k$，間隔 $h$ を適当に定め，分点

$$a=x_0 \quad x_1=a+h \quad x_2=a+2h \quad \cdots \quad x_n=a+nh=b$$

における関数値 $y_0$，$y_1$，$y_2$，…，$y_n$ を計算し，各 $y_i$ に $c$ を掛け $k$ を加えてプリンターに出力すべき空白の箇数 $m_i$ を算出し，改行して $m_i$ 箇の空白を出力して適当な記号（たとえば＊印）を出力すればよい。

〔プログラム例〕

```
 10 DEF FNC(X)=1/(1+X*X)      利用者が定義する
 20 PRINT "A,B,C,H,K="
 30 INPUT  A,B,C,H,K
 40 LET N=INT((B-A)/H+0.1)
 50 FŌR I=0 TŌ N
 60   LET X=A+I*H
 70   LET Y=FNC(X)
 80   LET L=INT(C*Y+0.5)+K
 90   FŌR J=1 TŌ L
100     PRINT "␣";
110   NEXT J
120   PRINT "*"
130 NEXT I
140 END
```




なお，上記の文90～文120のかわりに，

```
105     PRINT TAB(L);"*"
```

としてもよい（TABについては51ページ参照）。

**演習** 1 左記のプログラムの係数 $c$ を自動的に算定する方法を工夫せよ。上記のプログラムの $k$ についても考えよ。

2 グラフの上端と目盛りを出力するにはどうすればよいか（$c$ と $k$ の値は固定してよい。たとえば $c=10$，$k=20$）。

### 3.3.3 等　高　線

関数$f(x, y)$の等高線（等値線）を描かせたい場合がよくある。等高線を「線」として描くには，かなり手間がかかる。しかし，右下の出力例のような図ならば容易に描ける。そのプログラムを作ってみよう。

描く範囲は，$a \leqq x \leqq b$，$c \leqq y \leqq d$とし，関数$f(x, y)$の最小値と最大値のだいたいの値は$f_0$，$f_1$であるとする。分割の細かさは，$x$方向，$y$方向とも40分割，$f$の値に関しては10段階（しまの本数としては5本）とする。

〔プログラム例〕

```
 10 DEF FNC(X)=SIN(X)*CŌS(X)      利用者が
 20 PRINT "A,B,C,D,F0,F1="        定義
 30 INPUT  A,B,C,D,F0,F1
 40 LET H=(B-A)/40                x の間隔
 50 LET K=(D-C)/40                y の間隔
 60 LET W=(F1-F0)/10              z の間隔
 70 FŌR J=0 TŌ 40
 80   LET Y=C+(40-J)*K
 90   FŌR I=0 TŌ 40
100     LET X=A+I*H
110     LET Z=FNC(X,Y)
120     LET T=(Z-F0)/W
130     FŌR M=0 TŌ 10 STEP 2
140       IF T<M   THEN 170
150       IF T<M+1 THEN 190
160     NEXT M
170     PRINT "␣";
180     GŌTŌ 200
190     PRINT "*";
200   NEXT I
210   PRINT                       改行
220 NEXT J
230 END
```

〔**解説**〕 上下方向を$y$軸，左右方向を$x$軸とする。40等分だから

全部で（上下方向）41行　　（$j=0$，1，…，40）

各行に（左右方向）41字　　（$i=0$，1，…，40）

となる。これを上から1行ずつ出力するのであるが，その際

最初に出力する行（$j=0$）が$y$の上限$d$に対応

⋮

最後に出力する行（$j=40$）が$y$の下限$c$に対応

となることに注意する（こうしないと図が上下逆になってしまう）。したがって，第$j$行，第$i$字めの$x$，$y$の値は，

$x_i=a+ih$　　ただし，$h=(b-a)/40$

$y_j=c+(40-j)k$　　ただし，$k=(d-c)/40$

となる。このような各点について，以下の処理を行なう。

1） 関数値$z=f(x_i, y_j)$を計算する。それから

$t=(z-f_0)/w$　　ただし，$w=(f_1-f_0)/10$

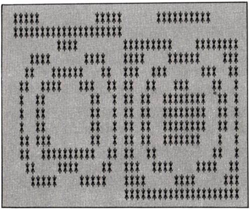
という変数変換を行なう。

2） $t$の値が，

$1\leqq t<2$，$3\leqq t<4$，$5\leqq t<6$，$7\leqq t<8$，$9\leqq t<10$

のどれかに入れば＊印，そうでなければ1字分の空白を出力。

上記2)の処理を行なっているのが，文130～文190である。



### 3.3.4 散布図（X-Y プロット）

データの組 $(x_1, y_1)$，$(x_2, y_2)$，…，$(x_n, y_n)$ を読み，$x$-$y$ 平面にプロットするプログラムを作ってみよう。

〔設計〕 記憶容量が十分にあれば，40×40ないし100×100程度の2次元配列を用意し，データをひとまずそこにプロットし，そこからまとめて出力する，という方法を用いるのが簡単でよい。

スケーリングの係数は，入力してもよいが，ここでは最大値と最小値の差をもとに算出することにする（こうすればスケール・オーバーの危険がない）。なお，座標軸のかわりに，平均値$\bar{x}$，$\bar{y}$を表す線を図に入れることにする。

〔プログラム例〕

```
 10 DIM L(41,41),X(100),Y(100)
 20 PRINT "N=";
 30 INPUT N
 40 PRINT "␣␣␣X␣␣␣␣␣␣␣␣Y"
 50 INPUT X(1),Y(1)
 60 LET X0=X(1)                 xの最小値の初期設定
 70 LET X1=X(1)                 xの最大値の    〃
 80 LET Y0=Y(1)                 yの最小値の    〃
 90 LET Y1=Y(1)                 yの最大値の    〃
100 LET S1=X(1)                 xの平均値の    〃
110 LET S2=Y(1)                 yの平均値の    〃
120 FOR M=2 TO N
130    INPUT X(M),Y(M)
140    LET S1=S1+X(M)
150    LET S2=S2+Y(M)
160    IF X(M)>X0 THEN 180
170    LET X0=X(M)
180    IF X(M)<X1 THEN 200
190    LET X1=X(M)
200    IF Y(M)>Y0 THEN 220
210    LET Y0=Y(M)
```

```
220     IF Y(M)<Y1  THEN 240
230     LET Y1=Y(M)
240  NEXT M
250  LET  H=(X1-X0)/40
260  LET  K=(Y1-Y0)/40
270  LET  I0=(S1/N-X0)/H
280  LET  J0=(S2/N-Y0)/K
290  FŌR  I=0  TŌ  40
300     FŌR  J=0  TŌ  40
310        L(I,J)=1
320     NEXT  J
330     LET  L(I,J0)=2
340  NEXT  I
350  FŌR  J=0  TŌ  40
360     LET  L(I0,J)=2
370  NEXT  J
380  FŌR  M=1  TŌ  N
390     LET  I=(X(M)-X0)/H
400     LET  J=(Y(M)-Y0)/K
410     LET  L(I,J)=3
420  NEXT  M
430  FŌR  J=40  TŌ  0  STEP  -1
440     FŌR  I=0  TŌ  40
450        ŌN  L(I,J)  GŌTŌ  460,480,500
460        PRINT  "␣";
470        GŌTŌ  510
480        PRINT  "+";
490        GŌTŌ  510
500        PRINT  "*";
510     NEXT  I
520     PRINT                              改行のため
530  NEXT  J
540  END
```

### 3.3.5 応用問題

**1　（ヒストグラム）**

N人の生徒の試験の点数（100点満点とする）を読み込み，

　　0～4点　　5～9点　　10～14点　　15～19点

　　20～24点　　……　　90～94点　　95点以上

に分けて人数を数え，点数の分布を棒グラフで出力するプログラムを作れ。

**2　（年齢別人口構成図）**

地方自治体の，男女別，年齢別の人口を読み込み，下記のように図化するプログラムを作れ。

```
90歳以上            *I**
80～89             **I***
70～79            ***I***
60～69          *****I******
50～59         ******I******
40～49        *******I*******
  …                ……
                 男 ←―― ――→ 女
```

**3　（関数のグラフ）**

3.3.2節で例示したプログラム（またはそれと同様な機能をもつプログラム）により，次の関数のグラフを描いてみよう。

i）$y=x^2$　　（$x=0$ から $x=1$ まで）

ii）$y=\sin x$　　（$x=0$ から $x=\pi$ ラジアンまで）

iii）$y=1/(1+x^2)$　　（$x=-3$ から $x=+3$ まで）

iv）$y=\sqrt{1-x^2}$　　（$x=-1$ から $x=+1$ まで）

v）$y=\cosh x$　　（$x=-2$ から $x=+2$ まで）

また，プログラムを少し改造して

$$y=1/x \quad (|x|\leqq 1,\ |y|\leqq 1 \text{ の範囲で})$$

のグラフを描いてみよう。

**4 （極座標）**

極座標で表された関数

$$r = f(\theta)$$

のグラフを描くプログラムを作れ。

**《ヒント》** 3.3.4 節のプログラムを改造すればよい。そこで用いる $x$，$y$ の値と $r$，$\theta$ の関係は

$$x = r\cos\theta$$

$$y = r\sin\theta$$

である。

**5 （模様を描く）**

上のプログラムを用いて，次の曲線を描いてみよう。

i ）うずまき

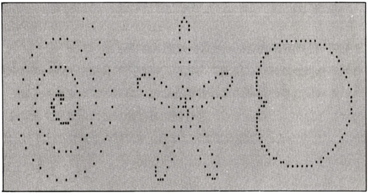
$r = \theta \qquad (0 \leqq \theta \leqq 6\pi)$

ii ）花模様

$r = \sin 5\theta \qquad (0 \leqq \theta \leqq 2\pi)$

iii ）心臓形（cardioid）

$r = \cos\theta + 1 \qquad (0 \leqq \theta \leqq 2\pi)$



## 3.4 行列(マトリックス)の計算

### 3.4.1 行列演算機能をもつ処理系の場合

TSS－BASICなど，一部の（主として中～大型コンピューターの）BASICには行列演算機能（MAT命令）を備えているものがある†。これを用いると，行列やベクトルの演算が次のように簡単に書ける。

| 機　能 | 書きかた（例） | 備　考 |
|---|---|---|
| 行列の和 | MAT C=A+B | 一般の式は書けない。 |
| 行列の差 | MAT C=A－B | |
| 行列の積 | MAT C=A＊B | |
| スカラー倍 | MAT C=(X)＊A | ( )内はスカラー |
| 逆行列 | MAT C=INV(A) | 式の中には書けない。(代入のみ) |
| 転置行列 | MAT C=TRN(A) | |
| 零行列 | MAT C=ZER | |
| 単位行列 | MAT C=IDN | |
| 行列の入力 | MAT READ A(3,3) | 入出力文の一種 |
| 行列の出力 | MAT PRINT A,B | |

特に「行列型」というような型があるわけではなく，DIM文で宣言した1次元配列がベクトル，2次元配列が行列として扱われる。

### 3.4.2 行列演算機能をもたない処理系の場合

行列演算機能は確かに便利であるが，処理系の負担が大きいので省略している機種も多い。そのような場合には，自分でプログラムを書かなければならない。といっても，逆行列以外は簡単で，定義式をそのままBASICの文になおして書けばよい。なお，逆行列の計算法についてはあとで説明する（4.8節)。

† BASICの元祖，本家のDartmouth大学のシステムには，第3版（1966年）から入っており，BASICの特長の一つになっていた。

**〔行列の乗算と入出力のプログラム例〕**　行列A，Bを入力し，その積C＝A・Bを計算して出力するプログラムの例を示す。Aは $l$ 行 $m$ 列，Bは $m$ 行 $n$ 列とする。（Cは当然，$l$ 行 $n$ 列となる）。

```
 10 DIM A(8,8),B(8,8),C(8,8)
 20 INPUT L,M,N
 30 FŌR I=1 TŌ L
 40   FŌR J=1 TŌ M
 50     PRINT "A(";I;",";J;")=";
 60     INPUT A(I,J)
 70   NEXT J                          Aの入力
 80 NEXT I
 90 FŌR I=1 TŌ M
100   FŌR J=1 TŌ N
110     PRINT "B(";I;",";J;")=";
120     INPUT B(I,J)
130   NEXT J                          Bの入力
140 NEXT I
150 FŌR I=1 TŌ L
160   FŌR J=1 TŌ N
170     LET S=0                       積の計算
180     FŌR K=1 TŌ M
190       LET S=S+A(I,K)*B(K,J)
200     NEXT K
210   LET C(I,J)=S
220   NEXT J
230 NEXT I
240 FŌR I=1 TŌ L
250   FŌR J=1 TŌ N                    Cの入力
260     PRINT "C(";I;",";J;")="
270   NEXT J      (前行の続き) ;C(I,J)
280 NEXT I
290 END
```

### 3.4.3　応用問題

MAT 文を使用できる機種の利用者はそれを用い，MAT 文を使用できない機種の利用者は MAT 文を使わずに書くこと。

**1　(行列の和)**

3行3列の行列A，Bを入力し，その和C＝A＋Bを出力するプログラムを作れ。

**2　(行列の積)**

コンピューターで次の計算をし，行列の乗算では交換法則が成立しないことを確認せよ。

$$\begin{bmatrix} 3 & 2 \\ 5 & 1 \end{bmatrix}\begin{bmatrix} 2 & 9 \\ 6 & 5 \end{bmatrix} \qquad \begin{bmatrix} 2 & 9 \\ 6 & 5 \end{bmatrix}\begin{bmatrix} 3 & 2 \\ 5 & 1 \end{bmatrix}$$

**3　(長方形行列)**

コンピューターで次の計算をし，結果を比較せよ。

$$\begin{bmatrix} 8 & 1 & 5 \\ 2 & 9 & 3 \end{bmatrix}\begin{bmatrix} 6 & 3 & 3 & 5 \\ 3 & 2 & 5 & 1 \\ 1 & 0 & 9 & 5 \end{bmatrix}$$

$$\begin{bmatrix} 6 & 3 & 1 \\ 3 & 2 & 0 \\ 3 & 5 & 9 \\ 5 & 1 & 3 \end{bmatrix}\begin{bmatrix} 8 & 2 \\ 1 & 9 \\ 5 & 3 \end{bmatrix}$$

(ヒント)　107ページのプログラムを利用してよい。

**4　(転置行列との積)**

行列A，Bを入力し，Aの転置行列とBの積

$$C = A^{T}B$$

を計算して出力するプログラムを作れ。また，

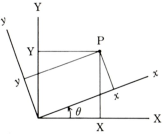
$$D = AB^{T}$$

を計算して出力するプログラムを作れ。これらのプログラムに，入力データとして特にAとBに同じ行列（長方形行列でもよい）を入れてみて，$A^{T}A$ および $AA^{T}$ が対称行列になることを確認せよ。

**5 (行列とベクトルの積)**

行列Aとベクトル$\boldsymbol{x}$を入力し，その積$\boldsymbol{y}=\mathrm{A}\boldsymbol{x}$を出力するプログラムを作れ。

**6 (座標変換)**




右の図の X-Y 座標系で（X，Y）と表される点の $x$-$y$ 座標（$x$，$y$）は次の式で求められる。

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} \mathrm{X} \\ \mathrm{Y} \end{bmatrix}$$

$\theta$およびX，Yを入力して$x$，$y$の値を計算して出力するプログラムを作れ。

**7 (連乗)**

行列A，Bとベクトル$\boldsymbol{x}$（行列は$n$行$n$列，ベクトルは$n$次元）を入力し，積

$$\boldsymbol{y}=\mathrm{AB}\boldsymbol{x}$$

を計算して出力するプログラムを作れ。

**《ヒント》** 先に積ABを計算し，それを$\boldsymbol{x}$に掛ける，という手順で計算すると計算量が多くなって損である。それよりも，先にB$\boldsymbol{x}$を計算し，それにAを掛ける方がよい。

**8 (要素パターンの出力)**

ディスプレイ装置やプリンターは横幅が限られているので，列の数が多くなると，行列の1行分を出力の1行におさめることが困難になる。そういう場合，詳しい値を見るためには，何行にも分けて出力するほかないが，デバッグなどの目的には

要素が0ならば「.」印

要素が正ならば「+」印

要素が負ならば「−」印

というような形式で全体のパターンを見たいことがある。そのためのプログラムを作れ。

## 3.5 擬似乱数による数値実験

### 3.5.1 関数 RND

統計的現象の「実験」をコンピューターで行なうことができる。そのための道具として RND という関数が用意されている。

関数 RND は，$0 \leqq u < 1$ の範囲の一様乱数を作り出す。すなわち，関数 RND を呼ぶたびに，上記の範囲の値 $u$ がランダム（random, 無作為，不規則，でたらめ）に出て来る（一様分布に従って）。

実際にどんな数が出て来るのか，次のようなプログラムを作って，実行させてみよう。

**乱数列を出力するプログラム**

```
10 FŌR I=1 TŌ 20
20    LET U=RND
30    PRINT U
40 NEXT I
50 END
```

出力例

```
.591065
.207991
.550967
.635863
.10411
.806334
4.29073E-02
.953862
.354289
.556498
.445778
1.31287E-02
.543443
.531664
.316403
.466435
.102909
.979818
.144224
.28198
```

**乱数発生のしくみ** 乱数を作るには，サイコロを振るように，物理現象を用いるのが理想的であるが，コンピューターの中でそう簡単に良質の物理乱数を作ることはできないので，普通は，たとえば

$$N_{i+1} = (N_i \text{ を K 倍して M で割った余り})$$

$$u_{i+1} = N_{i+1} / M$$

（ただし，N，M，Kはいずれも正の整数）というような方法により，「乱数に似た性質をもつ数列」を作り出して使用する。これを擬似乱数という。これは規則的に作り出すのであるから，完全にランダムとはいえないが，うまく作れば（たとえば上記の方法の場合，定数K，Mと初期値 $N_0$ を適切に選べば）初等的数値実験の目的には十分使える，ということが知られている。

**RANDOMIZE 文** 擬似乱数は，数列としては乱数に近い性質をもつが，同じ初期値 $N_0$ を使えば同じ数列が出てしまう。それを避けるにはプログラムの最初に RANDŌMIZE と書けばよい。

### 3.5.2 サイコロに変換

関数 RND で得られるのは $0 \leqq u < 1$ の範囲の一様乱数だけであるが，これを適当に変換することにより，上記以外の分布に従う乱数を作ることができる。

たとえば，これを「1から6の目が等確率で出るサイコロ」の目の実現値に換算するには，

$0 \leqq u < 1/6$　ならば ⚀

$1/6 \leqq u < 2/6$　ならば ⚁

…

$5/6 \leqq u < 1$　ならば ⚅

とすればよいであろう。BASIC でこのような変換を行なうには，次のように書くと簡単でよい。

```
LET S=INT(6*U)+1
```

乱数の発生と変換を兼ねて次のように書くことができる。

```
LET S=INT(6*RND)+1
```

〔**使用例**〕 サイコロを2回ずつ 100 組（計 200 回）振って，2回の目の和がちょうど7になる組数を数えてみる。

```
 10 RANDŌMIZE
 20 LET N=0                                カウンターのリセット
 30 FŌR K=1 TŌ 100
 40    LET S1=INT(6*RND)+1                 } サイコロを
 50    LET S2=INT(6*RND)+1                 } 2回振る
 60    IF S1+S2<>7 THEN 80                 和≠7 ?
 70    LET N=N+1                           カウント・アップ
 80 NEXT K
 90 PRINT N;
100 END
```

**演習 1**　女の子が3人続けて生まれる確率を，数値実験によって調べるプログラムを作れ。

### 3.5.3 関数 RND の応用問題

**1 （算数練習機）**

0から9までの乱数を二つ作り（I，Jとする），

Iの値　プラス印　Jの値　イコール印

を表示し，生徒に答を入力させ（それをKとする），

I＋J＝K　ならば　O印

I＋J≠K　ならば　X印

を出力するプログラムを作れ。

**《ヒント》** 0から9までの乱数を作るには

```
INT(10*RND)
```

とすればよい。

**〔拡張〕** 上記のプログラムに少し手を加えれば，次のような諸機能を付加することができる。

i）最初に生徒が1～4のキーを押し，

1→足し算　2→引き算　3→掛け算　4→割り算

の練習ができるようにする。

**《ヒント》** ON 文で分岐させるとよい。

ii）10問やらせて，点数（正答数）を出力する。

iii）IとJの桁数を指定できるようにする（掛け算や割り算の練習の際には，Iの桁数とJの桁数を別個に指定できることが望ましい）。

iv）3.1節のプログラムを組み合わせることにより，

最大公約数と最小公倍数

分数の計算

時分秒の計算

丸めの方法

などをさせる。

v）誤答の場合，もう一度同じ問題をやらせ，それでもだめならば正解を教えてやる。

**2 （いんちきなサイコロ）**

1～6の目の出る割合を等しくせず，たとえば

| 目の数 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| 割　合 | 15% | 10% | 30% | 10% | 20% | 15% |

になるようにするにはどうすればよいか。

**（ヒント）** 累積分布関数を作って逆補間する，というのが最も一般的な方法である。たとえば，上の例では，まず

| 目の数 | 1以下 | 2以下 | 3以下 | 4以下 | 5以下 | 6以下 |
|---|---|---|---|---|---|---|
| 割　合 | 15% | 25% | 55% | 65% | 85% | 100% |

という表を作ってコンピューターに記憶させ，0と1の間の一様乱数$u$の値が

0.15以下ならば ⚀
上記以外で0.25以下ならば ⚁
上記以外で0.55以下ならば ⚂
…

とすればよい。ただし，上の例のようにごく単純な比率の場合には，次のようにしてもよい（この方が速い）。

```
V=INT(20*RND)
```

Vの値が
- 0，1，2 ならば ⚀
- 3，4 ならば ⚁
- 5，6，7，8，9，10 ならば ⚂
- 11，12 ならば ⚃
- 13，14，15，16 ならば ⚄
- 17，18，19 ならば ⚅

具体的には，配列Mを用意し，

M(0)=1　M(1)=1　M(2)=1　M(3)=2
M(4)=2　M(5)=3　M(6)=3　…

というぐあいにセットしておいて，M(V)を目の数にするわけである。

### 3.5.4 正 規 乱 数

平均値0，分散1の正規分布$N(0,1)$に従う乱数$y$を作るには,「$0 \leqq u < 1$の一様乱数$u$を12箇加えて6を引く」という方法

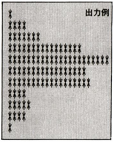
$$y = u_1 + u_2 + \cdots + u_{12} - 6$$

が最も簡単である。$y$が（近似的に）$N(0,1)$に従うことは，中心極限定理による。

**〔プログラム例〕** 上記の方法で正規乱数を100個作り，そのヒストグラムを描くプログラムを作ってみよう。ヒストグラムの範囲と間隔は，

$-3 \leqq y < +3$，　$\Delta y = 0.5$　とする。

```
 10 DIM H(12)
 20 RANDŌMIZE                         かきまぜる
 30 FŌR I=1 TŌ 12                   ┐
 40   LET H(I)=0                    │ カウンターをクリア
 50 NEXT I                          ┘
 60 FŌR K=1 TŌ 100                    100回反復
 70   LET Y=-6                      ┐
 80   FŌR J=1 TŌ 12                 │ 正規乱数を作る
 90     LET Y=Y+RND                 │
100   NEXT J                        ┘
110   LET I=INT(2*Y+6)
120   IF I<1  THEN 150
130   IF I>12 THEN 150
140   LET H(I)=H(I)+1
150 NEXT K
160 FŌR I=1 TŌ 12
170   IF H(I)=0 THEN 210
180     FŌR L=1 TŌ H(I)
190       PRINT "*";
200     NEXT L
210   PRINT
220 NEXT I
230 END
```


出力例

### 3.5.5 指数乱数

確率密度関数 $\lambda e^{-\lambda x}$ の指数分布（その平均値は $\lambda^{-1}$，分散は $\lambda^{-2}$ である）に従う乱数を作るには，区間［0，1］上の一様乱数 $u$ を

$$x=-\lambda^{-1}\log_e u$$

で変換すればよい。BASIC では，これを利用者定義関数の形

```
DEF  FNE=−M*LOG(RND)
```

にして用いると便利である（ただし，M$=\lambda^{-1}=$(平均値))。

〔用途〕　単位時間内に平均 $\lambda$ 人がランダムに窓口に到着するとき，その到着間隔（1人が来てから次の人が来るまでの時間）は上記の分布に従う。そのため，これは待ち行列のシミュレーションによく用いられる。

### 3.5.6 Poisson 乱数

単位時間内に平均 $\lambda$ 人がランダムに窓口に到着するとき，単位時間内に実際に到着する人数 $n$ の分布は，平均値 $\lambda$ の Poisson 分布

$$\mathrm{P}(n)=e^{-\lambda}\frac{\lambda^n}{n!}$$

に従う。このような乱数 $n$ を作るには，普通，次のようにする。

区間［0，1］上の一様乱数 $u_1$，$u_2$，$u_3$，…を作る。

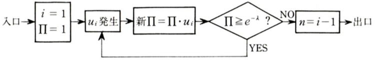
それを順に掛けてゆく。$u_1$，　$u_1u_2$，　$u_1u_2u_3$，　…

積が $e^{-\lambda}$ より小さくなったら止める。

その直前までの箇数を $n$ とする。すなわち

$$u_1u_2\cdots u_n\geqq e^{-\lambda}>u_1u_2\cdots u_{n+1}$$

〔解説〕　これは，3.5.5 節の方法で指数乱数（いいかえれば「到着間隔」）$x_1$，$x_2$，…を発生させて，その和が1（すなわち単位時間）になるまでの人数 $n$ を数えるのと本質的に同じことで，その「log の和」をとるかわりに「積の log」を用いたことに相当する。



### 3.5.7 待ち行列のシミュレーション

サービス窓口が一つあって，

客の到着する時間間隔は指数分布（母数 $\lambda$）に従う

サービスの所要時間は正規分布 $N(\mu, \sigma^2)$ に従う

という場合の数値実験を行なって，待ち行列の状態を調べてみよう。

**〔プログラムの設計〕** 各種の方式があるが，ここでは初心者のために，記憶場所を十分に使って，なるべく明快に書くことにする。主な変数は

| | | | |
|---|---|---|---|
| $A_i$ | $i$ 番目の客の到着時刻 | X | 到着間隔 |
| $T_j$ | $j$ 番目の客の終了時刻 | S | サービス所要時間 |
| $\overline{O}$ | サービス開始時刻 | Q | 行列の長さ |
| $i$ | 「次に到着する客」の番号 | $j$ | 「処理中の客」の番号 |

**準備** 前ページの方法により指数乱数Xを順に作って，到着時刻を

$$A_1 = 0 \qquad A_{i+1} = A_i + X \qquad (i = 1, 2, \cdots, n-1)$$

により計算して配列Aに入れ，「最初の客が到着した状態」

$$i = 1 \qquad j = 0 \qquad Q = 0$$

を初期値にして「到着」に進む。

**到着時の処理** 新 $i = i + 1$ とし，

$\begin{cases} Q = 0 \text{ ならば，} Q = 1,\ \overline{O} = A_{i-1} \text{ にして「サービス開始」に進む。} \\ Q \geqq 1 \text{ ならば，新} Q = Q + 1 \text{ にして「時計」に進む。} \end{cases}$

**サービス開始時の処理** 新 $j = j + 1$ とし，正規乱数Sを作り，

$$T_j = \overline{O} + S$$

により，サービス終了時点を算定して「時計」に進む。

**サービス終了時の処理** 今の客の待ち時間 $T_j - A_j$，現時点における待ち行列の長さQなどを出力し，Qから1を引き，

$\begin{cases} Q = 0 \text{ ならば「到着」に進む。} \\ Q \geqq 1 \text{ ならば，} \overline{O} = T_j \text{ として「サービス開始」に進む。} \end{cases}$

**時計**

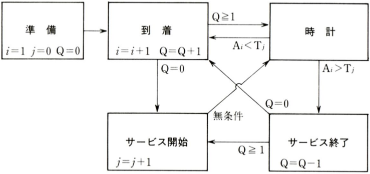
$\begin{cases} A_i \leqq T_j \text{ ならば「到着」に進む。} \\ A_i > T_j \text{ ならば「サービス終了」に進む。} \end{cases}$

〔プログラム例〕

```
 10 DIM A(100),T(100)
 20 LET N=100
 30 PRINT "LAMBDA,MU,SIGMA=";
 40 INPUT L,C1,C2
 50 LET A(1)=0
 60 FŌR I=1 TŌ N-1
 70   LET X=-LŌG(RND)/L
 80   LET A(I+1)=A(I)+X
 90 NEXT I
100 LET I=1
110 LET J=0
120 LET Q=0
130 REM   *** TŌUTYAKU ***
140 LET I=I+1
150 IF I>N THEN 420
160 IF Q<=0 THEN 190 ─┐
170 LET Q=Q+1         │
190 GŌTŌ 260          │               時計へ
190 LET Q=1  ←────────┘
200 LET Ō=A(I-1)
210 REM  *** SERVICE KAISI ***
220 LET J=J+1
230 GŌSUB 340                         正規乱数を作る
240 LET T(J)=Ō+S
250 REM     *** TŌKEI ***
260 LF A(I)<=T(J) THEN 140            到着へ
270 REM *** SERVICE SYURYŌU ***
280 PRINT A(J),T(J),T(J)-A(J),Q
290 LET Q=Q-1
300 IF Q<=0 THEN 140                  到着へ
310 LET Ō=T(J)
320 GŌTŌ 220                          サービス開始へ
```

```
330 REM   *** SEIKI RANSUU ***
340 LET Y=-6
350 FŌR K=1 TŌ 12
360    LET Y=Y+RND
370 NEXT K
380 LET S=C1+C2*Y
390 IF S>=0 THEN 410
400 LET S=0
410 RETURN
420 END
```

380行：$N(C_1,\ C_2^2)$ に換算




```
RUN
LAMBDA,MU,SIGMA
? 3,5,2
 0          3.51928    3.51928    11
 .175276    6.10783    5.93255    15
 .698696    9.03124    8.33255    20
 .89739     17.0419    16.1445    41
 1.04831    18.8557    17.8074    46
 1.80241    23.1985    21.3961    60
 1.87417    23.9396    22.0655    59
 2.92374    25.7708    22.847     64
 2.93948    29.5895    26.65      74
 3.28537    33.444     30.1586    86
```

**演習**　**1**　上記のプログラムを改造して，平均待ち時間（$T_j-A_j$ の平均値）を計算できるようにせよ。

# 4 数値計算

## 4.1 近似式

コンピューターで関数を扱う方法としては

近似式で計算する

数表を補間して用いる

の二つが基本的である。本節では前者について簡単に説明する。

### 4.1.1 Taylor 級数

周知のとおり，解析関数はその収束円内で **Taylor 級数**

$$f(a+h)=f(a)+\frac{h}{1!}f'(a)+\frac{h^2}{2!}f''(a)+\cdots+\frac{h^n}{n!}f^{(n)}(a)+\cdots$$

によって表すことができる。上式で特に $a=0$ とした形

$$f(h)=f(0)+\frac{h}{1!}f'(0)+\frac{h^2}{2!}f''(0)+\cdots+\frac{h^n}{n!}f^{(n)}(0)+\cdots$$

は **Maclaurin 級数**とも呼ばれている。

（例）

$$e^x=1+\frac{x}{1!}+\frac{x^2}{2!}+\cdots+\frac{x^n}{n!}+\cdots$$

$$\sin x=\frac{x}{1!}-\frac{x^3}{3!}+\frac{x^5}{5!}-\cdots+(-1)^k\frac{x^{2k+1}}{(2k+1)!}+\cdots$$

$$\cos x=1-\frac{x^2}{2!}+\frac{x^4}{4!}-\cdots+(-1)^k\frac{x^{2k}}{(2k)!}+\cdots$$

コンピューターでは無限項の計算はできないから，これを有限項で打ち切って計算することになるが，上記の三つの級数は収束が非常に速いので，たとえば $|x|\leqq 1$ の範囲で用いる場合，$e^x$ で10項，$\sin x$，$\cos x$ で5項もとれば誤差を $10^{-6}$ 以下にすることができる。

### 4.1.2　近似式の改良

前ページに例示したような，非常に収束の速い級数もあるが，一方には収束が非常に遅い級数がある。

（例）

$$\arctan x = x - \frac{x^3}{3} + \frac{x^5}{5} - \cdots + (-1)^k \frac{x^{2k+1}}{2k+1} + \cdots$$

$$\log(1+x) = x - \frac{x^2}{2} + \frac{x^3}{3} - \cdots + (-1)^{k+1} \frac{x^k}{k} + \cdots$$

このような級数で，たとえば $x=1$ の付近で誤差 $10^{-6}$ 以下になるようにするには，50万項ぐらいとらなければならない。コンピューターがいかに速いといっても，これでは実用に向かない。そのような場合

{ 加速手法（補外法）を併用する
{ 近似式を改良する

の二つの方法がある。前者は，だいたいどんな関数にも容易に適用できるので，自作のプログラムで無限級数の計算をする場合などに適している。それに対し後者は，高度な技術と経験を必要とし，個々の関数の特性を考慮して行なう必要があるので，専門家向きであるが，うまく作られた近似式の計算時間は，前者の計算時間にくらべて断然短い。

よく使われる関数については，既に詳しく研究され，近似公式集が作られている（次ページの文献リスト参照）。また，そのプログラムが作られて，組み込み関数として，あるいは技術計算ライブラリーとして提供されているから，調べて活用するとよい。参考までに，このような近似式の二，三の例を紹介する。

**常用対数の簡単な近似式の一例（Hastings の公式）**

$$\log_{10} x \cong c_1\left(\frac{x-1}{x+1}\right) + c_3\left(\frac{x-1}{x+1}\right)^3$$

$$c_1 = 0.86304 \qquad c_3 = 0.3645$$

適用範囲　$\frac{1}{\sqrt{10}} \leqq x \leqq \sqrt{10}$　　最大誤差　約 $5 \times 10^{-4}$

**tan $x$ の近似式の一例（永坂の公式）**

$$\tan\frac{\pi}{4}x \cong \left(0.13451446+\frac{2.6252949}{4.0333293-x^2}\right)x$$

適用範囲　$-1 \leqq x \leqq 1$　　最大誤差　$2.5\times10^{-6}$

**誤差関数の近似式の一例（Hastings の公式）**

$$\frac{2}{\sqrt{\pi}}\int_0^x e^{-t^2}dt \cong 1-\frac{1}{(1+a_1x+a_2x^2+a_3x^3+a_4x^4)^4}$$

$a_1=0.278393$　　$a_3=0.000972$

$a_2=0.230389$　　$a_4=0.078108$

適用範囲　$0 \leqq x < \infty$　　最大誤差　$5\times10^{-4}$

**Bessel 関数の近似式の一例（Allen の公式）**

$$J_0(x) \cong a_0+a_1\left(\frac{x}{3}\right)^2+a_2\left(\frac{x}{3}\right)^4+\cdots+a_6\left(\frac{x}{3}\right)^{12}$$

$a_0=1$　　$a_1=-2.2499997$　　$a_2=1.2656208$

$a_3=-0.3163866$　　$a_4=0.0444479$

$a_5=-0.0039444$　　$a_6=0.0002100$

適用範囲　$-5 \leqq x \leqq 3$　　最大誤差　$10^{-7}$

〔**文献**〕　各種の関数の近似式が次の文献にある。

1）山内，宇野，一松（編）；電子計算機のための数値計算法III，培風館（1972）
2）一松　信；初等関数の数値計算，教育出版（1974）
3）一松　信；近似式，竹内書店（初版1963，改訂版1972）
4）ヘイスティングス（竹内訳）；電子計算機のための近似計算法，東京図書（1973）
5）J. F. Hart; *Computer approximation,* Wiley（1968）
6）Y. L. Luke; *The special functions and their approximations,* I, II. Academic Press（1969）
7）磯田和男，大野　豊（編）；数値計算ハンドブック，オーム社（1971）

### 4.1.3　応　用　問　題

**1　（有限 Taylor 級数による sin $x$ の近似）**

119ページに示した sin $x$ の展開式の

第1項のみによる近似式　$f_1(x)=x$

第2項までによる近似式　$f_2(x)=x-\frac{x^3}{6}$

第3項までによる近似式　$f_3(x)=x-\frac{x^3}{6}+\frac{x^5}{120}$

…　　　　…

のグラフを描き，sin $x$ のグラフと比較せよ。

**《ヒント》**　プロットする範囲は $0 \leqq x \leqq 5$，間隔は $\varDelta x=0.1$ とすれば十分であろう。数値を出力して方眼紙にプロットしてもよいが，3.3.2 節のプログラムを利用してもよい。

**2　（誤差曲線）**

120ページの公式で $\log_{10} x$ の値を計算するプログラムを作り，

$0.3 \leqq x \leqq 3.0$　　（間隔 0.1）

の範囲でその誤差を計算してグラフを描いてみよう（BASIC の組み込み関数 LOG(X) の値を信用することにする。ただしそれは自然対数の値であるから，LOG(10) で割って常用対数になおした上で比較すること）。

**3　（巧妙な近似式と平凡な計算式の比較）**

121ページの永坂の公式のプログラムを作り，他の方法，たとえば

$\tan\theta=\sin\theta/\cos\theta$

$\tan\theta \fallingdotseq \theta+\frac{1}{3}\theta^3+\frac{2}{15}\theta^5+\frac{7}{315}\theta^7$

と精度および計算時間を比較せよ。

**《ヒント》**　精度は $\theta=0$ から $\theta=\pi/4$ まで $\varDelta\theta=\pi/40$ 間隔（すなわち 0° から 45° まで 5° おき）に調べればよいであろう。計算時間は1回だけの実行では時間が短すぎてうまく計測できないので，

```
FŌR I=1 TŌ 1000
    tan θ の計算式
NEXT I
```

というように多数回反復させて計るとよい。

**4 （誤差関数）**

121 ページの近似式により誤差関数を計算するサブルーチンを作れ。

**5 （0 次の Bessel 関数）**

121 ページの近似式により，$J_0(x)$ を計算するサブルーチンを作れ。

**6 （ガンマ関数）**

ガンマ関数 $\Gamma(x)$ の値を計算するには，まず，公式

$$\Gamma(x+1)=x\Gamma(x)$$

をくりかえし使って引数を $1 \leqq x \leqq 2$ の範囲に移し，Hastings の近似式（31番）

$$\Gamma(1+x)=1+a_1x+a_2x^2+a_3x^3+a_4x^4+a_5x^5$$

$$a_1=-0.5748646 \qquad a_2=0.9512363$$

$$a_3=-0.6998588 \qquad a_4=0.4245549$$

$$a_5=-0.1010678$$

を用いるのが比較的簡単である。

（例）　$\Gamma(3.85)$ を計算するには，まず

$$\Gamma(3.85)=\Gamma(2.85+1)=2.85\Gamma(2.85)$$

$$\Gamma(2.85)=\Gamma(1.85+1)=1.85\Gamma(1.85)$$

と変形する。これで引数が 1 と 2 の間に入ったから，$x=0.85$として上記の近似式に入れて $\Gamma(1+0.85)$ の値を計算し，その結果に1.85と2.85を掛ければよい。

このような方法で $\Gamma(x)$ の値を計算するサブルーチンを作れ。

**7 （ベータ関数）**

上で作ったガンマ関数のプログラムを利用して $B(p, q)$ の値を計算するサブルーチンを作れ。公式は，$B(p, q)=\dfrac{\Gamma(p)\Gamma(q)}{\Gamma(p+q)}$。

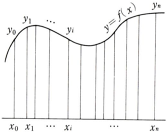
## 4.2 補　　間

近似式を作れば，計算時間が短く，取り扱いにも便利であるが，近似式の作成自体にかなりの手間がかかる。それに対し，数表を記憶させて補間して用いると，処理時間の点では多少不利になるが，とにかく簡単で，どんな関数でも近似できるので便利である。数表を記憶させるといっても，コンピューターの場合は必要に応じて高次の補間公式を使うことができるから，筆算用の数表よりも粗い（独立変数の間隔の広い）ものでよく，普通の目的には数十点（変動の多い関数でも数百点）のデータで十分である。数表による方法は，いわゆる任意関数（設計者がフリーハンドで描いた曲線，実験的に得られた特性曲線など）の表現に特に適しており，よく用いられている。

### 4.2.1 表の引きかた

関数 $y=f(x)$ の数表を

| | |
|---|---|
| $x_0$ | $y_0$ |
| $x_1$ | $y_1$ |
| $x_2$ | $y_2$ |
| ⋮ | ⋮ |
| $x_n$ | $y_n$ |




とする。引数（独立変数の値）が与えられたとき，まず，$x_l \leqq x < x_{l+1}$ になるような $l$ をさがす必要がある。それには次のようにすればよい。

**分点 $x_i$ が等間隔の場合**　　分点間隔を $h$ とすれば（すなわち

$$x_i = x_{i-1} + h \qquad (i=1,\ 2,\ \cdots,\ n)$$

とすれば），$l$ は次のようにして求められる。

$$l = 「(x - x_0)/h \text{ の整数部分}」$$

BASIC で書けば次のようになる。

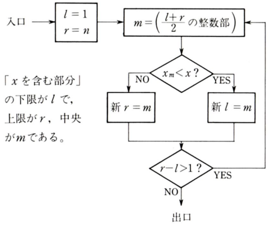
```
LET L=INT((X-X(0))/H)
```

**不等間隔の場合**　最も簡単な方法は，$i=0, 1, 2, \cdots$ の順に

$$x_i \leqq x < x_{i+1}$$

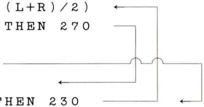
か否かを調べる（実際には右の不等式だけを調べれば十分）という方法であるが，もっと能率のよい方法として，2分法というのがある。これは，まず引数を表の中央の値（$x_{n/2}$ または $x_{(n+1)/2}$）と比較して「表の上半分にあるか下半分にあるか」を判定し，次にその上半分または下半分をまた上下に2分して，そのどちらにあるかを調べる，…という方法である。

〔流れ図〕



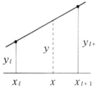

〔プログラム例〕（前後の処理は省略）

```
200 REM --NIBUNHŌU--
210 LET L=1
220 LET R=N
230 LET M=INT((L+R)/2)
240 IF X(M)<Z THEN 270
250 LET R=M
260 GŌTŌ 280
270 LET L=M
280 IF R-L>1 THEN 230
290 REM ------- EXIT ------
```



### 4.2.2 線 形 補 間

2点 $x_l$, $x_{l+1}$ における関数値 $y_l$, $y_{l+1}$ を1次式で補間するのが線形補間である。点 $x$ における関数値 $y$ は次のように（近似）計算できる。




$$y=y_l+\frac{y_{l+1}-y_l}{x_{l+1}-x_l}(x-x_l)$$

また次のように表すこともできる。

$$y=\frac{y_{l+1}-y_l}{x_{l+1}-x_l}x-\frac{x_l y_{l+1}-x_{l+1} y_l}{x_{l+1}-x_l}$$

### 4.2.3 Lagrange 補間

一般に，$m+1$ 箇の点 $x_0$, $x_1$, …, $x_m$ における関数値 $y_0$, $y_1$, …, $y_m$ が与えられたとき，

$$P(x_i)=y_i \quad (i=0,\ 1,\ \cdots,\ m)$$

となるような $m$ 次多項式を作って補間を行なうやりかたを Lagrange 補間という。このような多項式は，次のようにして作ることができる。

$$P(x)=y_0N_0(x)+y_1N_1(x)+\cdots+y_mN_m(x)$$

ただし

$$N_i(x)=\prod_{\substack{j=0\\j\neq i}}^{m}\frac{(x-x_j)}{(x_i-x_j)}$$

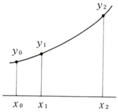
この $P(x)$ を Lagrange 補間多項式という。

（例）　$m=2$ の場合

$$N_0(x)=\frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)}$$

$$N_1(x)=\frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)}$$

$$N_2(x)=\frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)}$$




$$P(x)=y_0N_0(x)+y_1N_1(x)+y_2N_2(x)$$

〔解説〕　ここに用いた関数 $N_i(x)$ は

分点 $x_i$ における値は 1　　$N_i(x_i)=1$

他の分点 $x_j$ における値は 0　　$N_i(x_j)=0$　　$(j \neq i)$

という性質をもつ。実際，$N_i(x)$ の定義式に $x=x_i$ を代入すれば，すべての因子が約分できて $N_i(x_i)=1$ になり，$x=x_j(j \neq i)$ を代入すれば，分子に $(x-x_j)$ という因子があるから，$N_i(x_j)=0$ となる。この性質を利用して「点 $x_i$ における関数値だけを合わせる（他の分点には影響を与えずに）」ということができるわけである。

〔プログラム例〕（前後の処理は省略）

```
100 LET P=0
110 FOR I=0 TO M
120   LET N=1
130   FOR J=0 TO M
140     IF I=J THEN 160
150     LET N=N*(Z-X(J))/(X(I)-X(J))
160   NEXT J
170   LET P=P+Y(I)*N
180 NEXT I
```

〔注意〕　いま述べた方法を用いれば容易に高次の補間を行なうことができるが，次数を上げれば常に精度が良くなるとは限らず，次数を上げすぎるとかえって誤差が大きくなる。その一つの理由は，データの誤差が高次補間によって拡大されるためである。精密に計算された数表の補間には高次の式を用いてもよいが，実験データの補間には高次の式を使わない方がよい。なお，関数の性質と分点のとりかたによっては，データの誤差が 0 でも，次数を上げると誤差が無限に増大することがある。

（例）　関数 $f(x)=1/(1+x^2)$ を $-5 \leqq x \leqq 5$ の範囲で，等間隔分点によって補間すると，分点数を増すに従って誤差は限りなく増大する（**Runge の現象**）。証明と数値例は次の文献を見よ。

〔文献〕

1）ウェンドロフ(戸川訳)；理論数値解析，サイエンス社（1973）

2）森　正武；曲線と曲面，教育出版（1974）

### 4.2.4 応 用 問 題

**1　(等間隔の数表の補間)**

分点が等間隔な数表の線形補間を行なうプログラムを作れ。

**《ヒント》**　124ページに要領が書いてある。数表に関するデータ

$x_0 \quad h \quad n$

$y_0 \quad y_1 \quad y_2 \quad \cdots \quad y_n$

はDATA文で書いてREAD文で入力し，引数$x$はINPUT文により実行時点に入力できるようにするとよい。$x$を含む区間を知るには，124ページの最後の式を用いればよい。補間公式は，4.2.2節の2番目の式が簡単でよい。

**2　(不等間隔の数表の補間)**

分点が不等間隔の数表の線形補間を行なうプログラムを作れ。

**《ヒント》**　125ページのプログラムに，

数表の読み込み（$x$，$y$ともに必要）

補間の計算

結果の出力

のプログラムを書き加えればよい。

**3　(Lagrange 補間)**

Lagrange補間のプログラムを作れ。

**《ヒント》**　要するに127ページのプログラムに入出力の部分を書き加えればよいのであるが，数表の一部分だけ

$x_k \quad x_{k+1} \quad \cdots \quad x_{k+m}$

$y_k \quad y_{k+1} \quad \cdots \quad y_{k+m}$

を用いて計算できるようにするには，127ページのプログラムを少し改造する必要がある。また，数表のどの部分を用いるかを自動的に決めるようにするには，かなり複雑な処理を要する（$x$が数表の両端に近い場合が問題)。こういうプログラムは，あまり一般化しすぎると，複雑で使いにくいものになってしまう。自分で作るのなら，個々の仕事に合わせて，なるべく簡単なプログラムにするのが賢明である。

**4　(三角関数表を作る)**

よく知られている値

| $x$ | 0° | 30° | 45° | 60° | 90° |
|---|---|---|---|---|---|
| $\sin x$ | 0 | 1/2 | $\sqrt{2}/2$ | $\sqrt{3}/2$ | 1 |

を補間して，三角関数の数表（0°から 90°まで 5°間隔）を作り，厳密な値と比較せよ。

**5　(ガンマ関数表を作る)**

ガンマ関数は，普通，

$$\Gamma(x)=\int_0^\infty e^{-t}t^{x-1}dt$$

で定義されるが，

$$\Gamma(n+1)=n!$$

という性質があり，要するに「$n$ と $n!$ の関係をグラフ用紙にプロットして，その点を連続的に滑らかにつないだもの」であるから，

| $x=n+1$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | … |
|---|---|---|---|---|---|---|---|---|
| $n!$ | 1 | 1 | 2 | 6 | 24 | 120 | 720 | … |

を補間すれば，$\Gamma(x)$ の近似値を得ることができるであろう。この方法により，$x=1$ から $x=2$ まで，0.1 間隔のガンマ関数表を作り，厳密な値（小数点以下 5 桁までを下表に示す）と比較せよ。

| $x$ | $\Gamma(x)$ | $x$ | $\Gamma(x)$ |
|---|---|---|---|
| 1.0 | 1.00000 | 1.5 | 0.88623 |
| 1.1 | 0.95135 | 1.6 | 0.89352 |
| 1.2 | 0.91817 | 1.7 | 0.90864 |
| 1.3 | 0.89747 | 1.8 | 0.93138 |
| 1.4 | 0.88726 | 1.9 | 0.96177 |

**《ヒント》**　次数を上げすぎると失敗する。さりとて低次の補間で良い精度を望むのは無理である。より良い結果を得るには $x=1,\ 2,\ 3,\ 4$ における値を用いて $2\leqq x\leqq 3$ の範囲の値を求め，$\Gamma(x)=\Gamma(x+1)/x$ という公式で $1\leqq x\leqq 2$ の値に変換するのが一案であろう。

## 4.3 数値積分

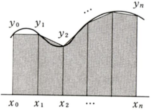
定積分 $S=\int_a^b f(x)\,dx$ の値を近似計算する方法について説明する。各種の公式があるが，利用者の立場からは，利用条件を

$$\begin{cases} f(x)\text{ が数表として与えられている場合} \\ f(x)\text{ が数式（またはプログラム）として与えられている場合} \end{cases}$$

に分けて，それぞれに適した公式を覚えておくとよいと思う。

### 4.3.1 被積分関数が数表で与えられている場合

分点 $x_0$，$x_1$，$x_2$，…，$x_n$ における関数値 $y_0$，$y_1$，$y_2$，…，$y_n$ が与えられているとする。分点 $x_0$，$x_n$ は積分の下限 $a$，上限 $b$ に一致しているとする。




**台形公式**　積分値 $S$ を台形の面積の和で近似する公式である。一般形は

$$S\cong\sum_{i=1}^{n}\frac{y_{i-1}+y_i}{2}\cdot(x_i-x_{i-1})=\frac{1}{2}\sum_{i=1}^{n}(y_{i-1}+y_i)(x_i-x_{i-1})$$

特に分点が等間隔（$x_i=x_{i-1}+h$,　$i=1,\ 2,\ \cdots,\ n$）ならば

$$S\cong\sum_{i=1}^{n}\frac{y_{i-1}+y_i}{2}\cdot h=\left(\frac{y_0}{2}+\sum_{i=1}^{n-1}y_i+\frac{y_n}{2}\right)\cdot h$$

となる。

**Simpson の公式**　等間隔の 3 点 $x_{2i}$，$x_{2i+1}$，$x_{2i+2}$ における関数値 $y_{2i}$，$y_{2i+1}$，$y_{2i+2}$ をもとに，被積分関数を 2 次式で近似して積分を行なう。この区間における積分値は

$$\int_{x_{2i}}^{x_{2i+2}}f(x)\,dx\cong\frac{y_{2i}+4\,y_{2i+1}+y_{2i+2}}{6}\cdot(x_{2i+2}-x_{2i})$$

これを $i=0$ から $i=n/2$ まで加え合わせれば $S$ の近似値が得られる（$n$ は偶数とする）。

**Newton-Cotes の公式**　これは上記の公式を一般化したもので，等間隔の $m+1$ 点における関数値 $y_{mi}$，$y_{mi+1}$，…，$y_{mi+m}$ をもとに，被

積分関数を$m$次式で近似して積分を行なう。この区間における計算式は，一般に次の形に書ける。

$$\int_{x_{mi}}^{x_{mi+m}} f(x)dx \cong \frac{c_0 y_{mi} + c_1 y_{mi+1} + \cdots + c_m y_{mi+m}}{C} h$$

ただし，$h = x_{mi+m} - x_{mi}$。係数は次のようになる。

| $m$ | $C$ | $c_0$ | $c_1$ | $c_2$ | $c_3$ | $c_4$ | $c_5$ | $c_6$ |
|---|---|---|---|---|---|---|---|---|
| 2 | 6 | 1 | 4 | 1 | | (Simpson の公式) | | |
| 3 | 8 | 1 | 3 | 3 | 1 | | | |
| 4 | 90 | 7 | 32 | 12 | 32 | 7 | | |
| 5 | 288 | 19 | 75 | 50 | 50 | 75 | 19 | |
| 6 | 840 | 41 | 216 | 27 | 272 | 27 | 216 | 41 |

〔Simpson の公式のプログラム例〕

```
10 DEF FNC(X)=1/(1+X*X)
20 INPUT A,B,N
30 LET H=(B-A)/(2*N)
40 LET S1=0
50 LET S2=0
60 FOR I=1 TO N-1
70 LET X=A+(2*I-1)*H
80 LET S1=S1+FNC(X)
90 LET X=A+2*I*H
100 LET S2=S2+FNC(X)
110 NEXT I
120 LET S1=S1+FNC(B-H)
130 LET S=(FNC(A)+4*S1+2*S2+FNC(B))*H/3
140 PRINT S
150 END
```

**公式の選びかた**

1）分点間隔が十分に細かければ，台形公式が安全確実。

2）分点間隔が粗い場合は，粗さに応じて高次の公式を使う。

3）分点間隔が粗くても台形公式の有利な場合がある。それは，

解析的周期関数の1周期上の積分　　解析関数の無限積分

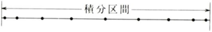
4）滑らかでない関数に高次の公式を用いるのは無意味。

### 4.3.2 被積分関数値をプログラムで計算できる場合

この場合には分点位置を自由に選べるので，より能率のよい計算を行なうことができる。合理的な分点のとりかたとしては，たとえば次のような方法が考えられる。

第１案　積分区間の中央付近は粗く，両端では細かくとる。

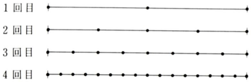
（例）




等間隔にするよりも，上記のようにする方が，補間多項式による近似の精度が良くなることが知られている。

第２案　粗い分割から始めて，分割をだんだん細かくしてゆく。

（例）

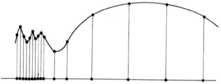



（以 下 同 様）

こうすれば，必要最小限の分点数で計算することができそうである。またあとで述べる補外法による精度向上も期待できる。

第３案　被積分関数の性質を調べ，滑らかな部分では粗く，変動の激しい部分は細かくする。

（例）



第１案を具体化したのが**Gauss 型の積分公式**で，次ページに挙げる Legendre-Gauss の公式はその代表的なものである。

第２案を具体化したものでは **Romberg 法**が有名である。

第３案を自動的に行なう計算法は「**適応型公式**」と呼ばれ，近年，各種の優れた公式が開発されている。

**Legendre-Gauss の公式**　積分区間の上の Legendre 多項式の零点を分点にとった公式である。このようにすると，$n$ 箇の分点を使用するだけで $2n-1$ 次までの多項式の厳密（離散化誤差 0）な積分値を得ることができる。この公式で計算するには，下記の表から分点位置 $x_1$，$x_2$，…，$x_n$ を算定し，そこにおける被積分関数値を計算し，表の係数を掛けて加え合わせ，区間幅を掛ければよい。

$$\int_a^b f(x)dx=(b-a)\sum_{i=1}^{n} C_i f(x_i)$$

ただし，$x_i=\dfrac{a+b}{2}+\dfrac{b-a}{2}t_i$

| | 分点番号 $i$ | 無次元化した分点位置 $t_i$ | 係数 $C_i$ |
|---|---|---|---|
| 2 点公式 | 1 | $-\sqrt{1/3}$ | 1/2 |
| | 2 | $+\sqrt{1/3}$ | 1/2 |
| 3 点公式 | 1 | $-\sqrt{3/5}$ | 5/18 |
| | 2 | 0 | 8/18 |
| | 3 | $+\sqrt{3/5}$ | 5/18 |
| 4 点公式 | 1 | −0.8611363116 | 0.173927422 |
| | 2 | −0.3399810436 | 0.326072577 |
| | 3 | +0.3399810436 | 0.326072577 |
| | 4 | +0.8611363116 | 0.173927422 |

〔**文献**〕　Gauss 型の積分公式の分点と係数の詳しい数表は下記 1)，2）にある。その他各種の公式については 3) が詳しい。

1）山内，宇野，一松；電子計算機のための数値計算法Ⅲ，培風館(1972)
2）A. H. Stroud and D. Secrest ; *Gaussian quadrature formulas,* Prentice-Hall（1966）
3）デービス，ラビノウィッツ(森訳)；計算機のための数値積分法，日本コンピュータ協会(1981)

### 4.3.3　応　用　問　題

**1　（簡単なテスト問題）**

数値積分により，次の値を求めよ。

$$(\text{i})\ \int_0^1 x^3\,dx \qquad (\text{ii})\ \int_0^1 e^x dx \qquad (\text{iii})\ \int_0^1 \frac{dx}{1+x^2}$$

**《ヒント》**　このような問題は，どの公式を用いても良い結果が得られる。高次の公式を使えば，( i )は厳密な値（丸め誤差の範囲内で）が得られ，( ii )についても粗い分割で十分良い精度が得られる。厳密な値は

( i )　$1/4=0.25$

( ii )　$e-1=1.718281828\cdots$

( iii )　$\pi/4=0.785398163\cdots$

**2　（対数表を作る）**

周知のとおり，$1/x$ の不定積分は $\log x$ で，$\log 1=0$ であるから，1 から $x$ まで $1/x$ を数値積分することによって $\log x$（の近似値）を求めることができるはずである。この方法により，$x=1$ から $x=2$ まで，$\Delta x=0.1$ 間隔の対数表を作成するプログラムを作れ。

**《ヒント》**　最初，$y(1)=0$ から出発して，

$$y(1.1)=y(1)+\int_{1.0}^{1.1}\frac{1}{x}dx$$

$$y(1.2)=y(1.1)+\int_{1.1}^{1.2}\frac{1}{x}dx$$

$$\cdots$$

を計算して出力すればよい。積分公式は，まず台形公式か Simpson の公式のような簡単なもので試みて，精度があまり悪いようならば高次の公式に切り換える。なお，上記のようにして得られるのは自然対数である。常用対数表を作るには，上で得られた値を $\log_e 10$ で割ればよい。

**3　（無限区間の積分）**

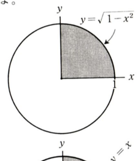
数値積分により $\int_0^\infty e^{-x^2}dx$ の値を計算せよ。

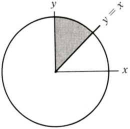
《ヒント》 無限区間の積分のためには，専用の公式（たとえば Laguerre-Gauss の公式など）もあるが，そういう特別な公式を使わなくても，被積分関数の値が十分に小さくなるまでとった有限区間の積分で近似しても，かなり良い精度が得られることが多い。いまの問題では，0 から 5 までを10等分して台形公式で積分する，という程度で十分な精度が得られる。厳密な積分値は

$$\sqrt{\pi}/2 = 0.88622692545\cdots$$

**4** （円の面積）

数値積分により $\int_0^1 \sqrt{1-x^2}\,dx$ の値を求めよ。




《ヒント》 これまでの例では容易に良好な結果が得られたが，この問題は「数値積分が常に簡単に成功するとは限らない」ということを示す教訓的難題であって，分割を細かくしても，高次の公式を使っても，なかなか思わしい結果が出ない。原因は，被積分関数の微係数が無限大になる点があるためである。このような積分をうまく計算するには，

IMT（伊理・森口・高澤）公式
DE（double exponential）公式
適応型の数値積分公式

などを用いる必要がある（いずれも高度な技法なので自作には向かないが，計算センターの数値計算ライブラリーなどに入っていれば利用するとよい）。なお，特異点を避けて，右下の図のように

$$2\int_0^{\sqrt{2}/2} (\sqrt{1-x^2}-x)\,dx$$

になおして計算すれば，そういう高等プログラムを使わなくても，普通の公式で容易に良い結果を得ることができる。

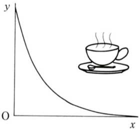
## 4.4 常微分方程式

コンピューターによる常微分方程式の計算法は，単独（非連立），1階の常微分方程式の初期値問題の解法が基本的で，それを応用すれば，連立の場合や高階の場合の計算も行なうことができる。

### 4.4.1 簡単な例題

はじめに，簡単な例題によって，基本的な考えかたを説明しておくことにする。

コップに100℃の湯を入れ，室温0℃の部屋に置く。湯の温度を$y$℃，時刻を$x$とする。湯の温度の変化率は（ごく単純化して考えれば）湯の温度と室温の差に比例するであろう。その比例定数は，コップの熱伝導率，表面積，比熱，その他いろいろなものに関係するが，ここでは話を簡単にするため，比例定数は1として計算することにする（一般の場合でも，いわゆる無次元化という処理をすれば同じ形になる）。以上の関係を式で書けば




微分方程式　$\dfrac{dy}{dx}=-y$　　　初期条件　$y(0)=100$

となる。このような関係を満たす関数$y(x)$を求める問題が，常微分方程式の初期値問題（の簡単な一例）である。

**簡単な解法**　上記の式の$dy/dx$を間隔$h$の差分商で代用すれば

$$\frac{y(x+h)-y(x)}{h}=-y(x)$$

移項すれば

$$y(x+h)=y(x)-h\cdot y(x)$$

となる。これを「現在時刻$t$における温度が既知のとき，その$h$時間後の温度$y(x+h)$を近似計算する式」として用いれば，初期値$y(0)$から出発して，$y(x+ih)$　($i=1, 2, 3, \cdots$)　を求めることができる。

〔プログラム例〕 時刻$x=1$までの計算を行なうことにする。

```
 10 INPUT H
 20 LET N=1/H              総ステップ数を計算
 30 LET X0=0              }
 40 LET Y=100             } 初期値の設定
 50 FŌR I=0 TŌ N
 60   LET X=X0+I*H        } 現時刻と現時点にお
 70   PRINT X,Y           } ける温度を出力
 80   LET Y=Y-H*Y          次の時点の温度を計算
 90 NEXT I
100 END
```

〔注意〕 上記のプログラムで実習する際には，ひとまず$h=0.1$で計算してプログラムが正常に動くことを確認し，次に$h=0.01$の場合を計算して，前の計算結果と比較するとよい。$h$をもっと小さくする場合には，上記のプログラムだと出力が多くなりすぎるので，次のようにして出力を間引く必要がある。

```
45 LET M=INT(0.1/H+0.5)
50 FŌR I=0 TŌ 10            }
60   LET X=X0+I*0.1         } 0.1間隔で出力
70   PRINT X,Y              }
75   FŌR J=1 TŌ M           }
80     LET Y=Y-H*Y          } 出力せずに進める
85   NEXT J                 }
90 NEXT I
```

なお，$h$は0.1の整数分の1にとるものとする。

〔補足〕 左記の計算式は，$y(x+h)=(1-h)y(x)$と書くこともできる。これより，

$$y(1)=y(x+nh)=(1-h)^n y(0)=100(1-1/n)^n$$

これの$n\to\infty$の極限値は，$100e^{-1}$となり，厳密解の値に一致する。

### 4.4.2 Euler 法

単独，1階の常微分方程式は，一般に

$$y'=f(x, y) \tag{1}$$

と書ける。ただし，$x$ は独立変数，$y(x)$ は未知関数，$y'$ は $dy/dx$ の意味，$f(x, y)$ は「$x$ と $y$ の値が与えられたときに $y'$ の値を求めるための計算式（必ずしも一つの式で表せなくてもよく，要するに $x$ と $y$ から $y'$ の値が一意的に計算できればよい）である。コンピューターで数値的に計算できるのは，一つの（与えられた）初期条件

$x=x_0$ のとき，$y=y_0$

から出発して，

$x_0$，$x_1$，$x_2$，…，$x_i$，…

（ただし，$x_i=x_{i-1}+h$，$h$ は分点間隔）におけ解 $y$ の近似値

$y_0$，$y_1$，$y_2$，…，$y_i$，…

を求めることである。このような計算をするには，基本的には，「1ステップ進めるための技術」すなわち「$x=x_i$ のときに $y=y_i$ である，ということがわかったときに，$x=x_{i+1}$ における $y$ の値 $y_{i+1}$ を計算する方法」を確立すればよい。

**〔計算の方法〕** 前小節と同様に，微分方程式(1)の微分商を差分商に置きかえれば

$$\frac{y(x+h)-y(x)}{h}=f(x, y)$$

これより

$$y(x+h)=y(x)+h\cdot f(x, y) \tag{2}$$

すなわち，

$$y_{i+1}=y_i+h\cdot f(x_i, y_i)$$

という計算法が導かれる。これを **Euler 法**という。

（例） 微分方程式 $y'+y=e^x$ の場合，$f(x, y)=e^x-y$ であるから，計算式は，$y_{i+1}=y_i+h(e^x-y)$ となる。

〔解説〕「微分を差分で代用する」というと乱暴なようであるが，関数$y$を点$x$のまわりのTaylor展開

$$y(x+h)=y(x)+\frac{h}{1!}y'(x)+\frac{h^2}{2!}y''(x)+\cdots+\frac{h^n}{n!}y^{(n)}(x)+\cdots$$

の２次以上の項を無視すれば式(２)が得られるし，式(１)を$x$から$x+h$まで積分して得られる関係式

$$y(x+h)=y(x)+\int_x^{x+h}f(x,\ y)dx$$

の被積分関数$f$を，定数（$x$における値）で近似しても式(２)が得られるわけで，さらに「分割を限りなく細かくするとき，Euler法の結果は厳密解に収束する」ということも証明されている(文献１，２参照)。

〔プログラム例〕

```
 10 INPUT X0,Y,H,N
 20 FŌR I=0 TŌ N
 30   LET X=X0+I*H
 40   PRINT X,Y
 50   GŌSUB 100
 60   LET Y=Y+H*F
 70 NEXT I
 80 STŌP
100 REM *** F(X,Y) NŌ KEISAN ***
110 LET F=X*Y                    f(x, y)の計算式の一例
120 RETURN
130 END
```

**演習**　1　次の微分方程式の数値解をEuler法で計算せよ。

(1)　$y'+xy=0$　　$y(0)=1$

(2)　$yy'=x^2$　　$y(0)=1$

(3)　$y'+y=\sin x$　$y(0)=1/2$

2　上のプログラムを改造して，解のグラフを出力できるようにせよ。また出力間隔を間引けるようにせよ。

### 4.4.3　Runge-Kutta 法

Euler 法は簡単で便利であるが，精度はあまりよくない。そこで，精度を改良した解法が数多く開発された。その中で，比較的簡単でありながら精度がよく安全確実な公式として最もよく使われているのが，以下で説明する Runge-Kutta 法である。

〔**計算の手順**〕　微分方程式は　$y'=f(x, y)$　とし，その $x=x_i$ における $y$ の値 $y_i$ までは計算済みとする。次の値（$x_{i+1}=x_i+h$ における $y$ の値 $y_{i+1}$）は，下記のようにして計算する。

1)　$x_i$ と $x_{i+1}$ の中点 $x_{中}=x_i+h/2$ における $y$ の値 $y_{中}$ を Euler法により計算する。

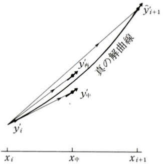
$$y_{中}=y_i+(h/2)y'_i$$

2)　それをもとに，中点における勾配 $y'_{中}$ を計算する。

$$y'_{中}=f(x_{中},\ y_{中})$$

3)　上記の $y'_{中}$ を用いて，中点のの値を再計算する。

$$y_{再}=y_i+(h/2)y'_{中}$$




4)　それをもとに，中点における勾配を再計算する。

$$y'_{再}=f(x_{中},\ y_{再})$$

5)　それを用いて，$x_{i+1}$ における $y$ の値の概算値 $\hat{y}_{i+1}$ を作る。

$$\hat{y}_{i+1}=y_i+hy'_{再}$$

6)　それをもとに，$y'_{i+1}$ の概算値 $\hat{y}'_{i+1}$ を求める。

$$\hat{y}'_{i+1}=f(x_{i+1},\ \hat{y}_{i+1})$$

7)　以上，4回計算された勾配 $y'_i$，$y'_{中}$，$y'_{再}$，$\hat{y}'_{i+1}$ を，1：2：2：1の重みで平均して用いて最終的な $y_{i+1}$ の値を計算する。

$$y_{i+1}=y_i+\frac{y'_i+y'_{中}+y'_{再}+\hat{y}'_{i+1}}{6}h$$

こうして計算される $y_{i+1}$ と真の $y(x_{i+1})$ の差は $h^5$ のオーダーである（要するに非常に小さい）ことが証明されている。

〔**数値例**〕 4.4.1節と同じ問題

$$y'=-y \qquad y(0)=100$$

を Runge-Kutta 法で $h=1$ として計算してみる。

$$x_{中}=0.5 \qquad y_{中}=100+(-100)/2=50 \qquad y'_{中}=-50$$

$$y_{再}=100+(-50)/2=75 \qquad y'_{中}=-75$$

$$\hat{y}_{i+1}=100-75=25 \qquad \hat{y}'_{i+1}=-25$$

$$y_{i+1}=100+\frac{-100-50\times2-75\times2-25}{6}\times1=37.5$$

この値は，$100e^{-x}$ の Taylor 展開式の第5項までの和の $x=1$ の値

$$100\left(1-\frac{1}{1!}+\frac{1}{2!}-\frac{1}{3!}+\frac{1}{4!}\right)=37.5$$

に等しい。

〔**プログラム例**〕 ここでは簡単のため†，関数 $f(x, y)$ が $x$ を含まない場合，すなわち微分方程式 $y'=f(y)$ の場合の例を示す。

```
 10 DEF FNC(Y)=   ……………………これは利用者が定義
 20 INPUT X0,Y,H,N
 30 FOR I=0 TO N
 40   PRINT X0+I*H,Y
 50   LET F1=FNC(Y)                    y'i
 60   LET F2=FNC(Y+H*F1/2)             y'中
 70   LET F3=FNC(Y+H*F2/2)             y'再
 80   LET F4=FNC(Y+H*F3)               ŷ'i+1
 90   LET F=(F1+2*F2+2*F3+F4)/6
100   LET Y=Y+F*H                      yi+1
110 NEXT I
120 END
```

† 基本 BASIC では関数の引数が1箇しか許されないので，$f(x, y)$ はサブルーチンの形で扱わなければならず，プログラムが長くなる。あとで示す「連立の場合のプログラム例」を参照されたい。

### 4.4.4　高階微分方程式の解法

$n$ 階微分方程式の場合，2階から $n-1$ 階までの導関数をそれぞれ別個の関数として扱えば，連立1階の微分方程式になる。

（例1）　$y''+2y'+3y=\cos x$　の場合，$y'$ を $z$ と置けば，

$$\begin{cases} y'=z \\ z'=y''=\cos x-2z-3y \end{cases}$$

（例2）　$y'''+yy''=0$　の場合，$y_1=y$，$y_2=y'$，$y_3=y''$ と置けば

$$\begin{cases} y_1'=y_2 \\ y_2'=y_3 \\ y_3'=-y_1 y_3 \end{cases}$$

このように連立1階の形にしてから，次小節で述べる方法を適用する。

**演習**　1　次の微分方程式を連立1階の形になおせ。

(1)　$y''+\sin y=0$　　(2)　$y''''+y=0$

(3)　$x^2y''+xy'+(x^2-1)y=0$

### 4.4.5　連立1階微分方程式の解法

連立1階の常微分方程式は，一般に次の形に書ける。

$$\begin{cases} y_1'=f_1(x,\ y_1,\ y_2,\ \cdots,\ y_n) \\ y_2'=f_2(x,\ y_1,\ y_2,\ \cdots,\ y_n) \\ \quad\cdots\cdots \\ y_n'=f_n(x,\ y_1,\ y_2,\ \cdots,\ y_n) \end{cases}$$

ただし，今回の添字は「変数の番号」を表している。

連立の場合の計算法は，本質的には単独の場合の計算法と同じで，ただそれを $n$ 箇の変数について同時に計算すればよい。

Euler 法を適用する場合の計算式は次のようになる。

$$\begin{cases} y_1(x+h)=y_1(x)+hf_1(x,\ y_1(x),\ \cdots,\ y_n(x)) \\ \quad\cdots\cdots \\ y_n(x+h)=y_n(x)+hf_n(x,\ y_1(x),\ \cdots,\ y_n(x)) \end{cases}$$

（例） 左記の例1の場合なら，次のように計算すればよい。

$$\begin{cases} y(x+h)=y(x)+hz \\ z(x+h)=z(x)+h(\cos x-2z-3y) \end{cases}$$

初期条件を $y(0)=0$，$z(0)=0$ とし，進み幅を $h=0.1$ とすれば

$$\begin{cases} y(0.1)=0+0.1\times 0=0 \\ z(0.1)=0+0.1\times(1-2\times 0-3\times 0)=0.1 \end{cases}$$

$$\begin{cases} y(0.2)=0+0.1\times 0.1=0.01 \\ z(0.2)\fallingdotseq 0.1+0.1\times(0.995-2\times 0.1-3\times 0)\fallingdotseq 0.1795 \end{cases}$$

以下同様。コンピューターで計算するのであれば，次のようにする。

```
10  INPUT X0,Y,Z,K,H        (ただしKは終了まで
20  FOR L=0 TO K             のステップ数)
30    LET X=X0+L*H
40    PRINT X,Y,Z
50    LET Y0=Y+H*Z
60    LET Z=Z+H*(COS(X)-2*Z-3*Y)
70    LET Y=Y0
80  NEXT L
90  END
```

Runge-Kutta 法を適用する場合の計算式は次のようになる。

$$y'_{i左}=f_i(x_{左},\ y_{1左},\ y_{2左},\ \cdots,\ y_{n左}) \qquad (i=1,2,\cdots,n)$$

$$y_{i中}=y_{i左}+(h/2)\,y'_{i左} \qquad (\ \ 〃\ \ )$$

$$y'_{i中}=f_i(x_{中},\ y_{1中},\ y_{2中},\ \cdots,\ y_{n中}) \qquad (\ \ 〃\ \ )$$

$$y_{i再}=y_{i左}+(h/2)\,y'_{i中} \qquad (\ \ 〃\ \ )$$

$$y'_{i再}=f_i(x_{中},\ y_{1再},\ y_{2再},\ \cdots,\ y_{n再}) \qquad (\ \ 〃\ \ )$$

$$y_{i右}=y_{i左}+hy'_{i再} \qquad (\ \ 〃\ \ )$$

$$y'_{i右}=f_i(x_{右},\ y_{1右},\ y_{2右},\ \cdots,\ y_{n右}) \qquad (\ \ 〃\ \ )$$

$$y_{i次}=y_{i左}+\frac{y'_{i左}+2y'_{i中}+2y'_{i再}+y'_{i右}}{6}h \qquad (\ \ 〃\ \ )$$

ただし，$x_{中}=x_{左}+h/2$，$x_{右}=x_{左}+h$，また $x_{右}$ と $y_{i次}$ が次のステップの $x_{左}$ と $y_{i左}$ になる。

〔プログラム例〕

```
 10 DIM Y(5),Y0(5),F(5)
 20 DIM F1(5),F2(5),F3(5)
 30 READ X0,N,H,K
 40 FŌR I=1 TŌ N
 50   READ Y0(I)
 60   LET Y(I)=Y0(I)
 70 NEXT I
 80 FŌR L=0 TŌ K
 90   LET X=X0+L*H
100   GŌSUB 500
110   FŌR I=1 TŌ N
120     LET F1(I)=F(I)
130     LET Y(I)=Y0(I)+F(I)*H/2
140   NEXT I
150   LET X=X+H/2
160   GŌSUB 500
170   FŌR I=1 TŌ N
180     LET F2(I)=F(I)
190     LET Y(I)=Y0(I)+F(I)*H/2
200   NEXT I
210   GŌSUB 500
220   FŌR I=1 TŌ N
230     LET F3(I)=F(I)
240     LET Y(I)=Y0(I)+F(I)*H
250   NEXT I
260   LET X=X+H/2
270   GŌSUB 500
280   FŌR I=1 TŌ N
290     LET Y(I)=Y0(I)+(F1(I)+
  (続き)  2*F2(I)+2*F3(I)+F(I))*H/6
300     LET Y0(I)=Y(I)
310     PRINT Y(I),
320   NEXT I
```

```
330    PRINT X          xの値は一番右の列に印刷される。
340  NEXT L
350  END
500  REM                fの計算のプログラムを利用者が書く
510  F(1)=Y(2)          (これは例1の計算の場合)
520  F(2)=COS(X)-2*Y(2)-3*Y(1)
530  RETURN
540  DATA 0,2,0.1,100
550  DATA 0             初期条件 { y(0) = 0
560  DATA 0                      { y'(0) = 0
570  END
```

### 4.4.6 常微分方程式の数値計算の注意

Runge-Kutta法は，1ステップ進む間（$x$から$x+h$までの間）各未知関数の4階までの導関数が連続であることを仮定している。したがって，その間に$f$の計算の不連続点を作らないように（たとえば，自動制御系のシミュレーションの場合，スイッチのON，OFFをしないように）注意する必要がある。

進み幅$h$は，小さすぎても大きすぎてもいけない。小さすぎると

$$y(x+h)=y(x)+(\text{増分})$$

の計算のとき，右図のようになって，増分が無視されてしまう。一方，進み幅が大きすぎると精度が悪くなり，さらに過大にすれば発散することがある。

```
     ×.×××××          y(x)
+  0.000000××         増分
―――――――――――――――――――――――――――
     ×.×××××          y(x+h)
   |←1ワード→|
```

〔文献〕

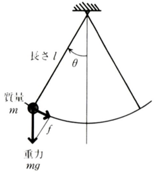
1) 一松 信；微分方程式と解法，教育出版(1977)

2) ヘンリチ(清水，小林訳)；計算機による常微分方程式の解法，I，II，サイエンス社(1973)

3) C. W. Gear；*Numerical initial value problems in ordinary differential equations,* Prentice-Hall (1971)

### 4.4.7 応用問題

**1 (大振幅の振子)**



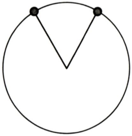

右図のような振子がある。おもりにかかる重力は $mg$ (ただし $g$ は重力加速度で約980 cm/ s$^2$), その接線方向成分 $f$ は, $f = mg\sin\theta$, したがって, トルクは $lf = lmg\sin\theta$, 一方, 質量慣性モーメントは $ml$ であるから, 運動方程式は

$$ml\frac{d^2s}{dt^2} = -lmg\sin\theta$$

ただし, $s$ は接線にそって計った変位であり, $\theta$ と $s$ の関係は (角度 $\theta$ の単位をラジアンとすれば) $s = l\theta$ である。この関係を入れると結局

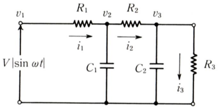
$$\frac{d^2\theta}{dt^2} = -\frac{g}{l}\sin\theta$$

これが, 大振幅の振子の微分方程式で, 初等関数の範囲では解けない。$l = 30$ cm, 初期条件 $\theta_0 = \pi/6$ ラジアン, 初速0として, $t = 0$ から10秒間の解を求めよ (Euler 法ならば $\Delta t$ を小さくとる必要がある。 Runge-Kutta 法ならば $\Delta t$ は0.1秒ぐらいでよい)。これがうまく解けたら, さらに大きな $\theta_0$ について計算してみるとよい (たとえば, $\theta_0 = (5/6)\pi$ ラジアン。右下の図参照)。



〔注意〕 $\theta_0$ を大きくすると周期が長くなるから, 解を求める範囲 (終端時刻) をそれに合わせて拡大する (遅らせる) 必要がある。一方, 進み幅は小さくするわけにいかない (支点の直下を通過する時の速度が大きいから)。そのため, $\theta_0$ を大きくすると計算所要時間が急増する。

**2 (C-R フィルター)**




右の図は，電源回路などに用いられるC-R回路の一例である。左側の端子に整流器から出てきた脈流を入れる。その電圧を

$$v_1 = V|\sin \omega t|$$

とする。端子電圧 $v_1$，$v_2$，$v_3$ の関係は，Ohm の法則により

$$i_1 R_1 = v_1 - v_2$$

$$i_2 R_2 = v_2 - v_3$$

$$i_3 R_3 = v_3 - 0$$

となる。またコンデンサーの性質から

$$\frac{dv_2}{dt} = \frac{i_1 - i_2}{C_1} \qquad \frac{dv_3}{dt} = \frac{i_2 - i_3}{C_2}$$

以上をまとめると，次のような微分方程式が得られる。

$$\begin{cases} \dfrac{dv_2}{dt} = \dfrac{1}{C_1}\left(\dfrac{v_1 - v_2}{R_1} - \dfrac{v_2 - v_3}{R_2}\right) \\[2ex] \dfrac{dv_3}{dt} = \dfrac{1}{C_2}\left(\dfrac{v_2 - v_3}{R_2} - \dfrac{v_3}{R_3}\right) \end{cases}$$

ただし，$v_1 = V|\sin \omega t|$

回路定数 $R_1$，$R_2$，$C_1$，$C_2$，負荷抵抗 $R_3$，電源電圧 $V$ および角速度 $\omega$，進み幅 $\Delta t$，計算打ち切り時刻 $t_{終}$ を読み込んで，上の微分方程式の数値解を計算するプログラムを作れ。初期条件は

$$t = 0 \text{ において } v_2 = v_3 = 0$$

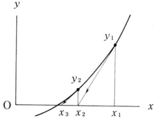
とする。

## 4.5　方程式 $f(x)=0$ の根

方程式 $f(x)=0$ の実現の計算によく使われる方法としては次のようなものがある。

**Newton 法**　適当な第1近似値 $x_1$ から出発して，

$$x_{k+1}=x_k-\frac{f(x_k)}{f'(x_k)}$$




($k=1, 2, \cdots$) によって精度を高めてゆく。上式は，点 $(x_k, y_k)$ において，曲線 $y=f(x)$ の接線を引き，$x$ 軸との交点を $x_{k+1}$ とすることを意味し，要するに $f(x)$ を局部的に1次式で近似して近似解を求めているわけである。根に近づくにしたがって，この近似の精度が良くなるため，収束はしだいに速くなる。適当な第1近似値がわかっている場合には非常に良い（速い）方法である。

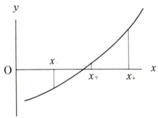
**〔プログラム例〕**　方程式 $x-\cos x=0$ を解くプログラムを作ってみよう。$f(x)=x-\cos x$，$f'(x)=1+\sin x$ であるから，

$$x_{k+1}=x_k-\frac{x_k-\cos x_k}{1+\sin x_k}$$

を反復すればよい。出発値は $x_1=1$ にとることにする。

```
10 LET X=1
20 FŌR K=1 TŌ 10
30    LET F=X-CŌS(X)
40    LET D=1+SIN(X)
50    LET X=X-F/D
60    IF ABS(F)<0.00001 THEN 80
70 NEXT K
80 PRINT "X=";X
90 END
```

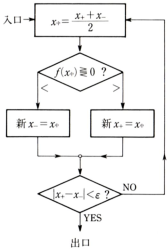
**2分法**　最初に，$f(x)>0$ になる点 $x_+$ と，$f(x)<0$ になる点 $x_-$ をさがしておく。次に $x_-$ と $x_+$ の中点 $x_中$ において関数値を計算し，その符号を調べる。もしこれが正ならば根は $x_-$ と $x_中$ の間にあり，負ならば根は $x_+$ と $x_中$ の間にあるから，それによって $x_+$ または $x_-$ を $x_中$ で置き換えて同様な計算を反復すればよい。




〔**プログラム例**〕　左記の例と同じ方程式 $x-\cos x=0$ の根を計算する場合のプログラム例を示す。出発値は

$$x_-=0 \qquad x_+=1$$

とする。根の存在範囲は反復1回ごとに 1/2 になってゆくから，10回反復すれば $|x_+-x_-|=2^{-10}\fallingdotseq 10^{-3}$ になる（精度はこの程度で実用上十分であろう）。




```
 10 LET M=0            (x−)
 20 LET P=1            (x+)
 30 FŌR K=1 TŌ 10
 40   LET X=(M+P)/2
 50   LET F=X-CŌS(X)
 60   IF F<0 THEN 90
 70   LET P=X
 80   GŌTŌ 100
 90   LET M=X
100 NEXT K
110 PRINT "X=";X
120 END
```

〔**文献**〕　上記以外の解法については次の文献が詳しい。

1)　戸川隼人；詳解数値計算演習，共立出版(1980)

## 4.6　高次代数方程式

本節では，実係数の $n$ 次代数方程式

$$x^n+a_1x^{n-1}+a_2x^{n-2}+\cdots+a_{n-1}x+a_n=0$$

のすべての根(実根，複素根とも)を計算するための Bairstow-Hitchcock 法について説明する。

**〔原理〕**　上記の方程式の左辺を $f(x)$ とする。これを，実係数の 2 次式 $x^2+px+q$ と，$n-2$ 次式

$$g(x)=x^{n-2}+b_1x^{n-3}+b_2x^{n-4}+\cdots+b_{n-3}x+b_{n-2}$$

の積に因数分解することを考える†。

$$f(x)=(x^2+px+q)\,g(x)$$

この分解ができれば，2 次方程式 $x^2+px+q=0$ を解いて二つの根が得られ，$g(x)$ の方はまた今と同様な因数分解をしてゆくことができるであろう。問題は，いかにして上記のような $p$，$q$ を求めるかであるが，Bairstow-Hitchcock 法では次のようにする。

まず，$p$ と $q$ の出発値（第 1 近似値）を適当に選ぶ。そして，2 次式 $x^2+px+q$ で $f(x)$ を割ってみる。その商を $\bar{g}(x)$，その係数を $b_1, b_2, \cdots$，$b_{n-2}$，余りを $rx+s$ とする。割り算の要領は

$$\begin{array}{r|l}
 & x^{n-2}+b_1x^{n-1}+\cdots+b_{n-2} \\
\hline
x^2+px+q & x^n+a_1x^{n-1}+a_2x^{n-2}+\cdots+a_{n-1}x+a_n \\
 & x^n+px^{n-1}\quad+qx^{n-2} \\
\cline{2-2}
 & (a_1-p)x^{n-1}+(a_2-q)x^{n-2}+a_3x^{n-3} \\
 & b_1x^{n-1}+b_1px^{n-2}+b_1qx^{n-3} \\
\cline{2-2}
 & \text{以下同様} \\
\cline{2-2}
 & \text{余り}\quad rx+s
\end{array}$$

計算式は右記 1) のとおりで，$r=b_{n-1}$，$s=b_n+pb_{n-1}$ となる。

---

† 実係数の代数方程式の複素根は必ず共役複素数になるので，このような実係数の因数分解が可能である。1 次式と $n-1$ 次式の積に分解しようとすると，一般に複素係数の多項式になり計算に手間がかかる。

$p$ と $q$ を与えると $r$ と $s$ が決まるのであるから，$r$ と $s$ は $p$ と $q$ の関数である。それが同時に 0 になるように，$p$，$q$ の値を修正する。修正量を $\Delta p$, $\Delta q$ とし 2 変数の Taylor 展開の 1 次の項までとるとすれば

$$\begin{cases} r(p+\Delta p,\ q+\Delta q) \fallingdotseq r(p,\ q) + \dfrac{\partial r}{\partial p}\Delta p + \dfrac{\partial r}{\partial q}\Delta q = 0 \\ s(p+\Delta p,\ q+\Delta q) \fallingdotseq s(p,\ q) + \dfrac{\partial s}{\partial p}\Delta p + \dfrac{\partial s}{\partial q}\Delta q = 0 \end{cases}$$

を解いて $\Delta p$, $\Delta q$ を決めればよい。その係数は次のようにして求める。

$$(x^2+px+q)\,\bar{g}(x)+(rx+s)=f(x) \qquad (\text{定義より})$$

$$\bar{g}(x)+(x^2+px+q)\frac{\partial g}{\partial q}+\frac{\partial r}{\partial q}x+\frac{\partial s}{\partial q}=0 \qquad \left(\begin{array}{c}\text{上式を } q\\ \text{で偏微分}\end{array}\right)$$

したがって，$g(x)\div(x^2+px+q)$ の余りの，$x$ の係数が $-\partial r/\partial q$ で定数項が $-\partial s/\partial q$。同様に $xg(x)\div(x^2+px+q)$ の余りの $x$ の係数が $-\partial r/\partial p$，定数項が $-\partial s/\partial p$ である。

**〔計算手順〕** 以上をまとめると，次のようになる。

1） $b_1=a_1-p \quad b_2=a_2-pb_1-q$

$i=3$，…，$n$ の順に

└ $b_i=a_i-pb_{i-1}-qb_{i-2}$

2） $c_1=b_1-p$, $c_2=b_2-pc_1-q$

$i=3$，…，$n-1$ の順に

└ $c_i=b_i-pc_{i-1}-qc_{i-2}$

3） 連立 1 次方程式

$$\begin{cases} c_{n-2}\Delta p+c_{n-3}\Delta q=b_{n-1} \\ (c_{n-1}-b_{n-1})\Delta p+c_{n-2}\Delta q=b_n \end{cases}$$

を解いて $\Delta p$, $\Delta q$ を求める。

4） 新 $p=p+\Delta p$, 新 $q=q+\Delta q$

（1）〜4）を $|r|$, $|s|$ が十分小さくなるまで反復。）

5） $x^2+px+q=0$ （その根がもとの方程式の根になる）

新 $a_i=b_i$ （$i=1$，2，…，$n-2$），新 $n=n-2$ として，手順 1 にもどる。ただし $n\leqq 2$ ならば直接解いて計算終了にする。

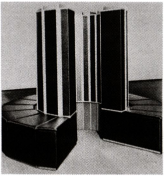
〔プログラム例〕

```
10 DIM A(10),B(10),C(10)
20 DIM X(10),Y(10)
30 INPUT N
40 FOR I=1 TO N
50 INPUT A(I)
60 NEXT I
70 LET E=0.001
80 LET K1=100
90 FOR M=N TO 3 STEP -2
100 FOR K0=1 TO 5
110 LET P=RND
120 LET Q=RND
130 FOR K=1 TO K1
140 LET B(1)=A(1)-P
150 LET B(2)=A(2)-P*B(1)-Q
160 FOR I=3 TO M
170 LET B(I)=A(I)-P*B(I-1)-Q*B(I-2)
180 NEXT I
190 LET C(1)=B(1)-P
200 LET C(2)=B(2)-P*C(1)-Q
210 IF M>3 THEN 240
220 LET A2=1
230 GOTO 280
240 FOR I=3 TO M-1
250 LET C(I)=B(I)-P*C(I-1)-Q*C(I-2)
260 NEXT I
270 LET A2=C(M-3)
280 LET A1=C(M-2)
290 LET A3=C(M-1)-B(M-1)
300 LET A4=C(M-2)
310 LET B1=B(M-1)
320 LET B2=B(M)
330 LET D=A1*A4-A2*A3
340 LET P1=(B1*A4-B2*A2)/D
350 LET Q1=(A1*B2-A3*B1)/D
360 LET P=P+P1
370 LET Q=Q+Q1
380 IF ABS(B1)+ABS(B2)<E THEN 420
390 NEXT K
400 NEXT K0
410 STOP
420 LET D=P*P-4*Q
```

```
430 IF D<0 THEN 490
440 LET X1=(-P+SQR(D))/2
450 LET X2=(-P-SQR(D))/2
460 PRINT X1
470 PRINT X2
480 GOTO 530
490 LET R=-P/2
500 LET S=SQR(-D)/2
510 PRINT R;"+";S;"I"
520 PRINT R;"-";S;"I"
530 IF M<=2 THEN 650
540 FOR I=1 TO M-2
550 LET A(I)=B(I)
560 NEXT I
570 NEXT M
580 IF N=INT(N/2)*2 THEN 610
590 PRINT -B(1)
600 GOTO 650
610 LET P=B(1)
620 LET Q=B(2)
630 LET M=2
640 GOTO 420
650 PRINT "OWARI"
660 END
```

**〔文献〕** 代数方程式の解法には，このほか，DKA 法，QD 法，Laguerre 法など，いろいろある。これらについては下記の文献を見よ。

1） 山本哲郎；数値解析入門，サイエンス社(1976)
2） ヘンリッチ(一松，平本，本田訳)；数値解析の基礎，培風館(1973)
3） 新谷尚義；数値計算 I，朝倉書店(1967)



スーパーコンピューターCRAY-1

**【4.5節と 4.6節の応用問題】**

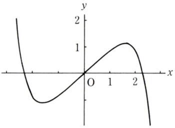
**1　（$f(x)=0$ の根）**

次の方程式を解け。

（i）　$2x-\tan x=0$

（ii）　$2x-\cosh x=0$

（iii）　$x^2-\sin x=0$




**$y=2x-\tan x$ のグラフ**

**《ヒント》**　このような問題の場合，まず，グラフを描いて，根の箇数や概略の値を調べておくとよい。たとえば(i)の場合，グラフは右の図のようになるから，根は3箇あり，その一つは $x=0$ で，他の二つは原点に関し対称であるから一方（たとえば正根）だけを求めればよく，その出発値としては $x=2$（あるいは2.1，2.2など）を用いればよい，ということがわかる。

(ii)の $\cosh x$ は，周知のとおり，双曲線余弦（hyperbolic cosine）で，定義は

$$\cosh x=(e^x+e^{-x})/2$$

導関数は

$$(\cosh x)'=\sinh x=(e^x-e^{-x})/2$$

である。

**2　（送電線）**

送電線のように自重でたれ下ったケーブルの形は，懸垂線（けんすいせん，catenary）になることが知られている。正確にいえば，最も低い点を原点とし，水平に $x$ 軸，垂直（上向きを正）に $y$ 軸をとるとき，曲線の方程式は

$$y=a(\cosh(x/a)-1)$$

となる。$a$ はパラメーターで，与えられた力学的（または幾何学的）条件から決まる。たとえば，鉄塔の間隔とケーブルの長さを与えた場合，$a$ は次のように計算できる。

原点から，ケーブル上の点（$x$，$y$）までの距離は

$$s=\int_0^x \sqrt{1+(y')^2}\,dx=a\sinh\frac{x}{a}$$

であるから，鉄塔の間隔を$l$，ケーブルの長さを$L$とすれば，

$$\frac{L}{2}=a\sinh\frac{l/2}{a}$$

この方程式を解けば$a$が得られる。

以上の関係を用いて，鉄塔の間隔が 200 m，ケーブルの長さ 220 mの場合の，たるみ（下図$w$）を求めよ。

〔ヒント〕 方程式

$$110=a\sinh(100/a)$$

を解くのであるが，そのまま計算しようとすると複雑になるので，

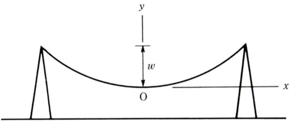
$$t=100/a$$

と置き，

$$1.1t=\sinh t$$

と変形した上で Newton 法などを適用するとよい。

〔文献〕 森口繁一；力学，日本機械学会（1949）




**3 （3次方程式，4次方程式，5次方程式）**

次の方程式を解け。

（i） $x^3-6x^2+11x-6=0$

（ii） $x^4-x^3-x^2-x-2=0$

（iii） $x^5+x^4-2x^3-2x^2+x+1=0$

## 4.7　連立1次方程式

**Gaussの消去法**　　連立1次方程式の計算には普通，Gaussの消去法が用いられる。要領は筆算とだいたい同じ。それを表の形で規則的手順により計算する。

筆算のやりかた

（例）
$$\begin{cases} 2x+4y+6z=6 \\ 3x+8y+7z=15 \\ 5x+7y+21z=24 \end{cases}$$

第1式を$x$について解くと

$$x=3-2y-3z$$

残りの式に代入して整理すると

$$\begin{cases} 2y-2z=6 \\ -3y+6z=9 \end{cases}$$

上の式を$y$について解くと

$$y=3+z$$

これを下の式に代入し整理すると

$$3z=18 \qquad \therefore\ z=6$$

順に上の式に代入して$y=9$，$x=-33$を得る

計算機による処理法

表にまとめる

| | | | |
|---|---|---|---|
| ② | 4 | 6 | 6 |
| 3 | 8 | 7 | 15 |
| 5 | 7 | 21 | 24 |

第1行を2で割り，3倍，5倍して第2行，第3行から引く。

| | | | |
|---|---|---|---|
| 1 | 2 | 3 | 3 |
| 0 | ② | −2 | 6 |
| 0 | −3 | 6 | 9 |

第2行を2で割り，−3倍して第3行から引く。

| | | | |
|---|---|---|---|
| 1 | 2 | 3 | 3 |
| 0 | 1 | −1 | 3 |
| 0 | 0 | ③ | 18 |

**一般の場合**　　係数を表にまとめる。

$$\begin{cases} a_{11}x_1+a_{12}x_2+\cdots+a_{1n}x_n=b_1 \\ a_{21}x_1+a_{22}x_2+\cdots+a_{2n}x_n=b_2 \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ a_{n1}x_1+a_{n2}x_2+\cdots+a_{nn}x_n=b_n \end{cases}$$

| | | | | |
|---|---|---|---|---|
| $a_{11}$ | $a_{12}$ | $\cdots$ | $a_{1n}$ | $b_1$ |
| $a_{21}$ | $a_{22}$ | $\cdots$ | $a_{2n}$ | $b_2$ |
| | | $\cdots\cdots$ | | $\cdots$ |
| $a_{n1}$ | $a_{n2}$ | | $a_{nn}$ | $b_n$ |

**〔計算手順〕**

前進消去

- $k=1,\ 2,\ \cdots,\ n-1$ の順に
  - 新 $a_{kj}=a_{kj}/a_{kk}$　　（$j=k,\ k+1,\cdots,n$）
  - 新 $b_k=b_k/a_{kk}$　　$j=k+1$からでもよい。
  - $i=k+1,\cdots,n$ について
    - 新 $a_{ij}=a_{ij}-a_{ik}a_{kj}$　　（$j=k,\ k+1,\cdots,n$）
    - 新 $b_i=b_i-a_{ik}b_k$
- $x_n=b_n/a_{nn}$

後退代入

- $k=n-1,\ n-2,\cdots,\ 1$ の順に
  - $x_k=b_k-\sum_{i=k+1}^{n}a_{kj}x_j$

```
10 DIM A(20,20),X(20),B(20)
20 INPUT N
30 FŌR I=1 TŌ N                        係数と定数項の読み込み
40   FŌR J=1 TŌ N
50     PRINT "A(";I;",";J;")=";
60     INPUT A(I,J)
70   NEXT J
80   PRINT "B(";I;")=";
90   INPUT B(I)
100 NEXT I
110 FŌR K=1 TŌ N-1                     ここから前進消去
120   FŌR J=K+1 TŌ N
130     LET A(K,J)=A(K,J)/A(K,K)
140   NEXT J
150   LET B(K)=B(K)/A(K,K)
160   FŌR I=K+1 TŌ N
170     FŌR J=K+1 TŌ N
180       LET A(I,J)=A(I,J)-
  (前行の続き)              A(I,K)*A(K,J)
190     NEXT J
200     LET B(I)=B(I)-A(I,K)*B(K)
210   NEXT I
220 NEXT K
230 LET X(N)=B(N)/A(N,N)
240 FŌR K=N-1 TŌ 1 STEP -1ここから後退代入
250   LET S=0
260   FŌR J=K+1 TŌ N
270     LET S=S+A(K,J)*X(J)
280   NEXT J
290   LET X(K)=B(K)-S
300   NEXT K
310   FŌR K=1 TŌ N
320   PRINT "X(";K;")=";X(K)
330 NEXT K
340 END
```

**ピボット選択**　前ページのプログラムには，一つ欠点がある。それは，文130および文150において，A（K，K）の値が0のとき，計算が不可能になることである。それを避けるためには，文110と文120の間で，次のような操作をすればよい。

$a_{kk}$　$a_{k+1,k}$　…　$a_{nk}$ の内で絶対値最大のものをさがす†

それが第 $l$ 行（$l \neq k$）にあれば第 $l$ 行と第 $k$ 行を交換する††

このような操作を**ピボット選択**という。ピボット（pivot）とは，消去する変数 $x_k$ の係数 $a_{kk}$ のことである。

〔**プログラム例**〕　前記のプログラムの文110の後に挿入

```
111  LET L=K                                  ↑
112  FŌR I=K+1  TŌ  N                         |
113    IF  ABS(A(I,K))<=ABS(A(L,K))           |
     (前行の続き)  THEN  115                  |
114    LET  L=I                     第k列の最大絶対値をさがす
115  NEXT  I                                  ↓
116  IF  L=K  THEN  125             l=kならばスキップ
117  FŌR  J=K  TŌ  N               ┐
118    LET  W=A(K,J)               │ 第k行と第
119    LET  A(K,J)=A(L,J)          │ l行を変換
120    LET  A(L,J)=W               │
121  NEXT  J                       ┘
122  LET  W=B(K)                   ┐
123  LET  B(K)=B(L)                │ b_k と b_l を交換
124  LET  B(L)=W                   ┘
125  FŌR  J=K+1  TŌ  N              (もとの文120)
```

† 脱線防止だけならば「0でないものをさがす」だけでよいが，絶対値最大のものをピボットに用いると丸め誤差を小さくする効果がある。

†† 「行の変換」は「連立方程式の式の順序の変換」に相当する。あとでその影響を補正する必要はない（列の交換をした場合は補正が必要）。

**掃出法 (Gauss-Jordan 法)**　係数と定数項を右のように表の形にまとめ，下記の手順で計算する。計算量はガウスの消去法より多いが計算手順が簡単なので小型計算機などで用いられる。

| | | | | |
|---|---|---|---|---|
| $a_{11}$ | $a_{12}$ | $\cdots$ | $a_{1n}$ | $b_1$ |
| $a_{21}$ | $a_{22}$ | $\cdots$ | $a_{2n}$ | $b_2$ |
| $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ |
| $a_{k1}$ | $a_{k2}$ | $\cdots$ | $a_{kn}$ | $b_k$ |
| $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ |
| $a_{n1}$ | $a_{n2}$ | $\cdots$ | $a_{nn}$ | $b_n$ |

$k=1,\ 2,\ \cdots,\ n$ の順に
- 第 $k$ 行を $a_{kk}$ で割る。
- $i=1,\ 2,\ \cdots,\ n$ について（ただし $i=k$ を除く）
  - 第 $k$ 行の $a_{ik}$ 倍を第 $i$ 行から引く。

計算終了時の $b_1$，$b_2$，…，$b_n$ の解が $x_1$，$x_2$，…，$x_n$ になっている。

〔**プログラム例**〕（入力の部分は前のページと同じなので省略）

```
110 FOR K=1 TO N
120   LET P=A(K,K)
130   FOR J=K TO N
140     LET A(K,J)=A(K,J)/P
150   NEXT J
160   LET B(K)=B(K)/P
170   FOR I=1 TO N
180     IF I=K THEN 240
190     LET Q=A(I,K)
200     FOR J=K TO N
210       LET A(I,J)=A(I,J)-Q*A(K,
220     NEXT J                (前行の続き) J)
230     LET B(I)=B(I)-Q*B(K)
240   NEXT I
250 NEXT K
260 FOR I=1 TO N
270   PRINT "X(";I;")=";B(I)
280 NEXT I
290 END
```

〔**文献**〕　連立1次方程式の計算技法に関しては下記の文献が詳しい。

1) 戸川隼人；マトリクスの数値計算，オーム社(1971)
2) ウェストレイク(戸川訳)；線形計算ハンドブック，培風館(1972)

## 4.8　逆　行　列

**定　義**　正方行列を$A$，単位行列を$I$とするとき，$AX=I$になる行列$X$を$A$の逆行列といい$A^{-1}$で表す。

**計算法（原理）**　逆行列を求めるには要するに

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdots & \cdots & & \cdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \cdots & \cdots & & \cdots \\ x_{n1} & x_{n2} & \cdots & x_{nn} \end{bmatrix} = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \cdots & \cdots & & \cdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}$$

になるような$x_{ij}(i,\ j=1,\ 2,\ \cdots,\ n)$を求めればよく，それには連立1次方程式　$A\boldsymbol{x}_j=\boldsymbol{e}_j$　（$\boldsymbol{x}_j$は$X$の第$j$列，$\boldsymbol{e}_j$は$I$の第$j$列）を$j=1,\ 2,\ \cdots,\ n$について解けばよい。

**計算法（実際）**　連立1次方程式の解法としては掃出法が簡単でよい（連立方程式を1題解くだけならばガウスの消去法の方が速いが，逆行列の計算の際には掃出法でもガウスの消去法でも計算時間は同じになる）。手順はまず

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $a_{11}$ | $a_{12}$ | $\cdots$ | $a_{1n}$ | 1 | 0 | $\cdots$ | 0 | 第1行 |
| $a_{21}$ | $a_{22}$ | $\cdots$ | $a_{2n}$ | 0 | 1 | $\cdots$ | 0 | 第2行 |
| $\cdots$ | $\cdots$ | | $\cdots$ | $\cdots$ | $\cdots$ | | $\cdots$ | |
| $a_{n1}$ | $a_{n2}$ | $\cdots$ | $a_{nn}$ | 0 | 0 | $\cdots$ | 1 | 第$n$行 |

という表を作り，

$k=1,\ 2,\ \cdots,\ n$の順に
　第$k$行を$a_{kk}$で割る
　$i=1,\ 2,\ \cdots,\ n$に対し（ただし$i=k$を除く）
　　第$i$行から第$k$行の$a_{ik}$倍を引く。

という計算をする。その終了時点に，表の左半分は$I$になり，右半分は$A^{-1}$になっている。

**記憶場所の節約**　上記の計算の中間結果は，第$k$段階の開始時点について考えてみると，配列Aの第$k$列より左側は単位行列と同じ形になっていて，実際にはもはや計算の必要はなく，また表の右半分の第$k$列

より右側は，実質的な計算はまだ行なわれておらず，単位行列の形がそのまま残っている。そこで，このように内容が自明な部分を除き，配列Aの第 $k$ 列より左には $A^{-1}$ を入れるようにすれば記憶場所が節約され，計算時間も半分で済む。次のプログラムは，そのような方式をとっている。

〔プログラム例〕

```
 10 DIM A(20,20)
 20 INPUT N                                   データ入力
 30 FŌR I=1 TŌ N
 40   FŌR J=1 TŌ N
 50     PRINT "A(";I;",";J;")";
 60     INPUT A(I,J)
 70   NEXT J
 80 NEXT I
 90 FŌR K=1 TŌ N                              ここから逆行列の計算
100   LET P=A(K,K)
110   LET A(K,K)=1
120   FŌR J=1 TŌ N
130     LET A(K,J)=A(K,J)/P
140   NEXT J
150   FŌR I=1 TŌ N
160     IF I=K THEN 220
170     LET Q=A(I,K)
180     LET A(I,K)=0
190     FŌR J=1 TŌ N
200       A(I,J)=A(I,J)-Q*A(K,J)
210     NEXT J
220   NEXT I
230 NEXT K
240 FŌR I=1 TŌ N                              結果の出力
250   FŌR J=1 TŌ N
260     PRINT "A(";I;",";J;")=",A(
270   NEXT J                   (前行の続き) I,J)
280 NEXT I
290 END
```

**【4.7 節と 4.8 節の応用問題】**

**1 （連立1次方程式）**

次の連立1次方程式を解け。

（i）
$$\begin{cases} 36x+30y+24z=3 \\ 30x+34y+26z=4 \\ 24x+26y+21z=4 \end{cases}$$

（ii）
$$\begin{cases} 6x+2y+3z=5 \\ 18x+14y+11z=21 \\ 24x+20y+16z=30 \end{cases}$$

**2 （逆行列）**

次の行列の逆行列を求めよ。

（i）
$$\begin{bmatrix} 2 & 4 & 6 \\ 3 & 8 & 7 \\ 5 & 7 & 21 \end{bmatrix}$$

（ii）
$$\begin{bmatrix} 68 & -41 & -17 & 10 \\ -41 & 25 & 10 & -6 \\ -17 & 10 & 5 & -3 \\ 10 & -6 & -3 & 2 \end{bmatrix}$$

（iii）
$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 2 & 2 & 2 \\ 1 & 2 & 3 & 3 & 3 \\ 1 & 2 & 3 & 4 & 4 \\ 1 & 2 & 3 & 4 & 5 \end{bmatrix}$$

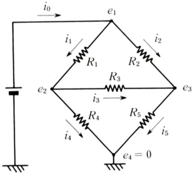
**3 （行列式）**

$n$ 行 $n$ 列の行列式の値を計算するプログラムを作れ。

《ヒント》 Gauss の消去法の計算をして，そのピボットを全部掛け合わせればよい（ただし，行の交換をした場合は，交換のたびに符号を逆にする）。

4　（ブリッジ回路）

右図の回路に電流から$i_0$アンペアの電流を図示の点に供給したときの各点の電位$e$および各素子に流れる電流$i$を計算するプログラムを作れ。




《ヒント》　Ohmの法則より

$$\left.\begin{array}{l} R_1 i_1 = e_1 - e_2 \\ R_2 i_2 = e_1 - e_3 \\ R_3 i_3 = e_2 - e_3 \\ R_4 i_4 = e_2 \\ R_5 i_5 = e_3 \end{array}\right\} \quad (1)$$

またKirchhoffの第1法則より

$$\left.\begin{array}{l} i_0 - i_1 - i_2 = 0 \\ i_1 - i_3 - i_4 = 0 \\ i_2 + i_3 - i_5 = 0 \end{array}\right\} \quad (2)$$

が成立する。式(1)を各$i$について解いて(2)に代入すれば

$$\begin{bmatrix} \dfrac{1}{R_1}+\dfrac{1}{R_2} & -\dfrac{1}{R_1} & -\dfrac{1}{R_2} \\ -\dfrac{1}{R_1} & \dfrac{1}{R_1}+\dfrac{1}{R_3}+\dfrac{1}{R_4} & -\dfrac{1}{R_3} \\ -\dfrac{1}{R_2} & -\dfrac{1}{R_3} & \dfrac{1}{R_2}+\dfrac{1}{R_3}+\dfrac{1}{R_5} \end{bmatrix} \begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} = \begin{bmatrix} i_0 \\ 0 \\ 0 \end{bmatrix}$$

となる。データとして

$R_1$　　$R_2$　　$R_3$　　$R_4$　　$R_5$　　$i_0$

を読み，連立方程式の係数を計算し，方程式を解き，結果を式(1)に入れて電流を計算して出力すればよい。

## 4.9 固有値，固有ベクトル

与えられた正方行列$A$に対し

$$A\boldsymbol{x}=\lambda\boldsymbol{x} \qquad \text{ただし } \boldsymbol{x}\neq\boldsymbol{0}$$

を満たすベクトル$\boldsymbol{x}$（**固有ベクトル**という）とスカラー$\lambda$（**固有値**という）を求める問題を，行列の固有値問題という。このような計算は，固有振動数の計算や主成分分析（統計解析の一手法）などの際に必要になる。$A$が実対称行列の場合とそうでない場合とでは，問題の性質も解法もかなり異なる。応用では実対称の場合が多く，実対称ならば計算が簡単なので，以下では実対称行列の固有値問題の解法について説明する。

**Jacobi 法**　実対称行列の固有値問題の解法は数多くあるが，最も簡単で，そのわりに能率がよく，安全確実なのは Jacobi 法である。

**〔原理〕**　実対称行列$A$の非対角要素の一つを$a_{pq}$とし，

$$M=\begin{bmatrix} \diagdown & & & & \\ & c & & s & \\ & & \diagdown & & \\ & -s & & c & \\ & & & & \diagdown \end{bmatrix} \begin{array}{l} \\ \leftarrow p\text{ 列目} \\ \\ \leftarrow q\text{ 列目} \\ \\ \end{array}$$

（対角上に 1 を並べる，他の部分は 0；$c$, $-s$ の列が $p$ 行目，$s$, $c$ の列が $q$ 行目）

ただし，

$$\theta=\frac{1}{2}\arctan\frac{2a_{pq}}{a_{pp}-a_{qq}}$$

$$c=\cos\theta$$

$$s=\tan\theta$$

という行列$M$を作って，新$A=MAM^{\mathrm{T}}$という変換をすると，

新 $a_{pq}=0$　　新 $a_{qp}=0$　　になる。

新$A$はやはり実対称で，$A$と同じ固有値をもつ。

という性質がある。このような変換をくりかえして，非対角要素をすべて 0（または非常に小さな値）にすれば，そのとき固有値は新$A$の対角上に並ぶ。また固有ベクトルは，変換に用いた$M^{\mathrm{T}}$の積

$$V=M_m^{\mathrm{T}}M_{m-1}^{\mathrm{T}}\cdots M_2^{\mathrm{T}}M_1^{\mathrm{T}}$$

の各列として得られる。$p$，$q$の選びかたはいろいろあるが，右のプログラムでは「その段階の非対角要素の中で絶対値最大のものを$a_{pq}$とする」という方式（古典的方法）を用いている。この方法は，最大値をさがすのに手間がかかるが，必要な反復回数は最も少ない。

〔**計算手順**〕 $V=I$（単位行列）として出発し，すべての非対角要素の絶対値が十分小さくなるまで，下記の手続きをくりかえす。

1） $p$，$q$ を決める（前ページ最後の説明参照）

2） $\theta=\dfrac{1}{2}\arctan\dfrac{2a_{pq}}{a_{pp}-a_{qq}}$ （ただし $a_{pp}=a_{qq}$ の場合は $\theta=\pi/4$ とする）

3） $c=\cos\theta \qquad s=\tan\theta$

4） 新$A=MAM^{\mathrm{T}}$ の乗算を行なう。実際には第 $p$ 行，第 $q$ 行，第 $p$ 列，第 $q$ 列の計算だけをすれば十分である（他の部分は不要）。

5） 新$V=VM^{\mathrm{T}}$ の乗算を行なう。実際には第 $p$ 列と第 $q$ 列の計算だけをすれば十分である（他の部分は不要）。

〔**プログラム例**〕

```
 10 DIM A(10,10),V(10,10)
 20 INPUT N
 30 FŌR I=1 TŌ N                          行列Aを読み込む
 40   FŌR J=1 TŌ N
 50     RPINT "A(";I;J;")=";
 60     INPUT A(I,J)
 70   NEXT J
 80 NEXT I
 90 FŌR I=1 TŌ N                          単位行列をVに入れる
100   FŌR J=1 TŌ N
110     LET V(I,J)=0
120   NEXT J
130   LET V(I,I)=1                        反復打ち切り回数
140 NEXT I                                      |
150 FŌR K=1 TŌ 1000 ←---------------------------┘
160 LET P=1                               最大値をさがす
170 LET Q=2
180 FŌR I=1 TŌ N-1
190   FŌR J=I+1 TŌ N
200     IF ABS(A(I,J))<=ABS(A(P,Q))
    (前行の続き)     THEN 230
210     LET P=I
```

```
220        LET Q=J
230     NEXT J
240   NEXT I
250   IF ABS(A(P,Q))<0.0001 THEN 500
260   LET B=A(P,P)-A(Q,Q)
270   IF B=0 THEN 300
280   LET T=0.5*ATN(2*A(P,Q)/B)
290   GŌTŌ 310                              θの計算
300   LET T=3.14159/4
310   LET C=CŌS(T)
320   LET S=SIN(T)
330   FŌR J=1 TŌ N
340     LET A1=A(P,J)
350     LET A2=A(Q,J)
360     LET A(P,J)= C*A1+S*A2
370     LET A(Q,J)=-S*A1+C*A2
380   NEXT J
390   FŌR I=1 TŌ N
400     LET A1=A(I,P)
410     LET A2=A(I,Q)
420     LET A(I,P)=C*A1+S*A2
430     LET A(I,Q)=-S*A1+C*A2
440     LET V1=V(I,P)
450     LET V2=V(I,Q)
460     LET V(I,P)= C*V1+S*V2
470     LET V(I,Q)=-S*V1+C*V2
480   NEXT I
490   NEXT K
500   FŌR J=1 TŌ N
510     PRINT "EV=",A(J,J)
520     FŌR I=1 TŌ N
530       PRINT V(I,J)
540     NEXT I
550   NEXT J
560   END
```

**【4.9 節の応用問題】**

**1　（プログラム・テストのための問題）**

次の行列の固有値と固有ベクトルを求めよ。

（i）
$$\begin{bmatrix} 5 & 4 & 1 & 1 \\ 4 & 5 & 1 & 1 \\ 1 & 1 & 4 & 2 \\ 1 & 1 & 2 & 4 \end{bmatrix}$$

（ii）
$$\begin{bmatrix} 1111 & -1089 & 891 & 909 \\ -1089 & 1111 & -909 & 891 \\ 891 & -909 & 1111 & -1089 \\ 909 & 891 & -1089 & 1111 \end{bmatrix}$$

**2　（弦の振動）**

弦の振動の方程式は，無次元化すると次の形に書ける。

$$-\frac{d^2u}{dx^2}=\lambda u$$

境界条件　$u(0)=u(1)=0$

この微分方程式の左辺を中心差分で近似すれば

$$\frac{-u_{i-1}+2u_i-u_{i+1}}{\Delta x^2}=\lambda u \qquad (i=1,\ 2,\ \cdots,\ n-1)$$

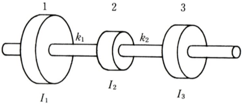
ただし，$u_i=u(x_i)=u(i/n)$　，$\Delta x=1/n$

行列の形で書けば

$$\begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & -1 & 2 & -1 & \\ & & \ddots & \ddots & \ddots \\ \text{空白の所は }0 & & & -1 & 2 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ \vdots \\ u_{n-1} \end{bmatrix} = \lambda \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ \vdots \\ u_{n-1} \end{bmatrix}$$

となる。$n$ を読み込み，上記の行列を作り，165 ページのプログラムを利用して固有値，固有ベクトルを計算して出力するプログラムを作れ。

3　（ねじり振動）




上の図のような系の，ねじり振動の固有振動数を計算するプログラムを作れ。ただし，軸まわりの慣性モーメント $I_1$，$I_2$，$I_3$ および（軸のねじりに関する）ばね定数 $k_1$，$k_2$ はデータとして入力するものとする。軸の両端は回転については拘束されていないものとする。

《ヒント》　円板1，2，3の回転角を $\varphi_1$，$\varphi_2$，$\varphi_3$ とすれば運動方程式は

$$I_1\ddot{\varphi}_1=k_1(\varphi_2-\varphi_1)$$
$$I_2\ddot{\varphi}_2=k_1(\varphi_1-\varphi_2)+k_2(\varphi_3-\varphi_2)$$
$$I_3\ddot{\varphi}_3=k_2(\varphi_2-\varphi_3)$$

この解を $\varphi_i(t)=a_i\sin pt$（$i=1, 2, 3$）と仮定して代入すれば

$$-p^2I_1a_1=k_1(a_2-a_1)$$
$$-p^2I_2a_2=k_1(a_1-a_2)+k_2(a_3-a_2)$$
$$-p^2I_3a_3=\qquad\qquad +k_2(a_2-a_3)$$

ここで

$$p^2=\lambda \quad x_1=\sqrt{I_1}\,a_1 \quad x_2=\sqrt{I_2}\,a_2 \quad x_3=\sqrt{I_3}\,a_3$$

と置いて整理すれば，行列の固有値問題

$$\begin{bmatrix} \dfrac{k_1}{I_1} & \dfrac{-k_1}{\sqrt{I_1}\sqrt{I_2}} & 0 \\ \dfrac{-k_1}{\sqrt{I_1}\sqrt{I_2}} & \dfrac{k_1+k_2}{I_2} & \dfrac{-k_2}{\sqrt{I_2}\sqrt{I_3}} \\ 0 & \dfrac{k_2}{\sqrt{I_2}\sqrt{I_3}} & \dfrac{k_2}{I_3} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \lambda \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

となる。入力データをもとに，この行列を作り，165ページのプログラムで $\lambda$ を求め，固有振動数 $f=\sqrt{\lambda}/(2\pi)$ を計算して出力すればよい。

# 付録　BASICからFORTRANへ

BASICでプログラミングを習った人が，研究や仕事の都合によりFORTRANを使用しなければならなくなる場合がよくある。その場合，FORTRANの教科書を勉強すれば，もちろんすぐ書けるようになるであろうが，既にBASICを知っているのだから，もっと近道の勉強法があるはずである。以下は，そのような近道の一つの試みである。

## A.1 概　　要

BASICとFORTRANはよく似ている。細かい仕様は違うけれども基本はだいたい同じである。たとえば，基本BASICの主な文はFORTRANの文と次のように対応している。

| 基本BASICの文 | FORTRANの文 | 説　明 |
|---|---|---|
| LET文 | 代入文 | 174ページ |
| FOR～NEXT文 | DO文 | 176 |
| IF文 | IF文 | 175 |
| ON文 | 計算型GOTO文 | 175 |
| GOTO文 | GOTO文 | 175 |
| GOSUB文 | CALL文 | 179 |
| RETURN文 | RETURN文 | 178 |
| IUPUT文 | READ文 | 180 |
| PRINT文 | WRITE文 | 180 |
| REM文 | 注釈行 | 174 |
| DIM文 | DIMENSION文 | 172 |
| STOP文 | STOP文 | 184 |

## A.2 データの型

BASICには，データの型として，数値型と文字列型があった。それに対し，FORTRANには次のような各種の型がある。

整数型　　実数型　　倍精度実数型　　複素数型

論理型　　文字型

**整数型と実数型の違い**　整数型というのは数値を固定小数点の整数の形で扱う型，実数型というのは数値を浮動小数点の形で扱う型のことである。前者は計算が簡単だから処理が速い。ことに添字（番地の修飾）に用いるには整数型が適している。後者は処理に時間がかかるけれども数値計算には便利である。

基本BASICの数値型は，数値をすべて浮動小数点の形で扱っているから，FORTRANの実数型に相当する。基本BASICで勉強した人にとって，整数型は全く新しいものであり，またFORTRANの文法において重要なものであるから，これを重点的に勉強する必要がある。

整数型で特に注意を要するのは次の2点である。

1）整数型の数の演算結果は整数型になり小数点以下は切り捨て

　（例）　2／3の結果は0

2）表現できる数値の範囲がせまい（普通，10桁ぐらいまで）

**文字型**　BASICの文字列型に相当するものは，FORTRAN 77にはあるけれど，普通のFORTRAN（水準7000）にはない。普通のFORTRANの文字型は，一つの変数に文字を4字しか入れることができず，代入や比較も許されず，BASICの文字列型よりはるかに不便なものである。

**その他**　倍精度実数型は，数値を倍精度浮動小数点の形で扱う型，複素数型は，実数型の値を二つ組にして実数部，虚数部を表し，複素数の演算を行なう型である。論理型は，論理代数（Bool 代数）の計算を行なう型である。

## A.3 変　数　名

6字までの英数字の組合せ（ただし先頭は英字に限る）を変数名として用いることができる。長い変数名を使えるから，なるべく物理的意味に即した名前を用いて，読み易いプログラムを書くとよい。

（例）　TAKASA　　HABA　　H2SŌ4

**変数名と型の関係**　FORTRANでは，型宣言をしないで変数名を用いた場合，変数名の第1字目（頭文字）が

I，J，K，L，M，Nならば整数型

上記以外（A～H，O～Z）ならば実数型

となる。型宣言をすれば，上記にかかわらず，自由に名前をつけることができるが，うっかり型を間違えたり型宣言を忘れたりすると困るから，なるべく上記の原則を守り（ことに1字だけを用いる場合），例外は最小限で済ませるようにすることを心掛けるとよい。

**型宣言の書きかた**　上記以外の場合は型宣言が必要である。型宣言は次のような宣言文による。宣言文はプログラムの冒頭に置く。

| | 標準的な書きかた | | 別の書きかた† |
|---|---|---|---|
| 整数型 | INTEGER | この右に変数名や配列名をコンマで区切って列記する | INTEGER*4 |
| 実数型 | REAL | | REAL*4 |
| 倍精度実数型 | DŌUBLE PRECISIŌN | | REAL*8 |
| 複素数型 | CŌMPLEX | | CŌMPLEX*8 |
| 論理型 | LŌGICAL | | LŌGICAL*4 |
| 文字型 | （なし） | | CHARACTER |

（例）　REAL　NAGASA，LAMBDA

**IMPLICIT文**　ある文字で始まる変数名をすべて同じ型にする場合には，次のように書くこともできる†。

IMPLICIT　型名（先頭の文字の列）

（例）　IMPLICIT　CŌMPLEX　（Z，C，W）

---

† これは使用できる機種と使用できない機種がある。

## A.4 配　　列

配列に関する機能は，基本的にはBASICとだいたい同じである。すなわち，宣言は

DIMENSIŌN 配列名(寸法)，配列名(寸法)，…

の形で書き，

配列名(添字)

の形で式の中などで引用することができる。次元は，普通，3次元まで許される。

〔注意事項〕

1） 添字は1から始まる。BASICと違って添字0が使えない†。

2） 添字は整数型に限る†。しかも，次の形以外は許されない†。

$c$　　$v$　　$v+c$　　$v-c$

$c*v$　　$c*v+c$　　$c*v-c$

ただし，$c$は符号なしの整数値，$v$は整数型の変数名である。

（例） A(I+2)　A(3*J−1)　などは良いが

A(2+I)　A(J*3)　などは許されない。

3） BASICだと，「10行10列以下ならば配列宣言を省略してもよい」という特例があったが，FORTRANではそれがなく，常に宣言を必要とする。

〔覚えておけば便利なこと〕

1） 型宣言と配列宣言をまとめて書くことができる。

（例） CŌMPLEX A(10,10),Z(8)

2） 共通領域（後述）の宣言と配列宣言を共用することもできる。

（例） CŌMMŌN P(3),C

† FORTRAN 77や，一部の拡張FORTRANでは，これらの制限は緩和されている。

## A.5 式

式の書きかたは BASIC とだいたい同じである。ただし下記の点に注意すること。

1）**べき乗**は記号＊＊で表す。

（例）　$x^y$ はX＊＊Y

2）**定数の型**は次のようにして区別される。

小数点も指数部もついてない数は整数型

小数点または指数記号Eがついていれば実数型

指数記号Dがついていれば倍精度実数型

（実数部，虚数部）の形をしていれば複素数型

．TRUE．および．FALSE．は論理型

| | | |
|---|---|---|
| （例） | 123 | は整数型 |
| | 123．45　や　1E6 | は実数型 |
| | 1．2D−3　や　1D6 | は倍精度実数型 |
| | （12．3，4．5） | は複素数型 |

なお，1E6 および 1D6 はどちらも $1\times10^6$ を表す。また 1．2D−3 は $1.2\times10^{-3}$ の意味。

3）**混合演算の禁止**　一つの式の中に，型の異なる変数をまぜて書くこと（これを混合演算という）は原則として禁止されている†。

（例）　Iが整数型，Rが実数型の場合，

I＋R　　　I＊R

と書くと誤りになる。また，定数1は整数型，1.0 は実数型だから

1＋R　は誤り。　1．0＋R　と書けばよい。

したがって，実数型の計算式に現れる定数には必ず小数点または指数部をつける必要がある。FORTRAN 77 や一部の拡張 FORT RANでは混合演算が許されているが，混合演算を行なうと型の変換のために計算時間が増加するから，なるべく避ける方が賢明である。

† ただし，$x^y$ の $x$ が実数型，$y$ が整数型というのは認められている。

## A.6 文

**文番号**　FORTRAN では，各文に行番号をつける必要がない。BASIC と同じようにしても誤りではないが，むだである。FORTRAN では，GOTO 文の行先のように，必要な文にだけ番号をつける。これを文番号という。文番号は小→大の順につける必要はなく，実行順序とも関係がない。

**コーディング・シートの書きかた**　FORTRAN の文は，書く位置が次のように制限されている。

各行の第 1 字目は「注釈か否か」を表す。ここが C ならば注釈。
文番号は第 1 字目～第 5 字目に書く。
第 6 字目は「前の行の続きか否か」を表す。ここが空白または 0 ならば「文の第 1 行目」である。
文の本体は第 7 字目～第 72 字目に書く。

文の本体

（例）

```
C ******** EXAMPME ******
     10 A=B+
       &                 C
        STŌP
```

A=B+C の意味

## A.7 代入文

FORTRAN の代入文は，BASIC の代入文の LET を省いたものと思えばよい。すなわち，=の右に書かれた式の値が計算されて，その結果が=の左に書かれた変数に代入される。

=の左と右の型は一致していなくてもよい。一致していない場合は，右辺の計算結果を左辺の変数の型に変換してから代入する。実数型や倍精度実数型の値を整数型に変換するときは，小数点以下が切り捨てられる。

## A.8 GOTO 文

**GOTO 文** BASIC と全く同じ。すなわち

GŌTŌ 行先の文番号

と書く。なお，GŌTŌのかわりに単にGŌと書いてもよい。

**計算型 GOTO 文** 次の形で書く。

GŌTŌ （行先$_1$，行先$_2$，…，行先$_n$），K

これは BASIC の ON 文に相当するもので，Kの値が 1，2，…，$n$ のとき，それぞれ行先$_1$，行先$_2$，…，行先$_n$ に進む。

## A.9 IF 文

これには算術 IF 文と論理 IF 文の 2 種類がある。

**算術 IF 文** これは次の形式で書く。

IF（算術式） 行先$_1$，行先$_2$，行先$_3$

算術式の値が負ならば行先$_1$へ，0 ならば行先$_2$へ，正ならば行先$_3$へ進む。

（例） IF（X－1．0） 20，30，30

$x < 1$ ならば文20へ，$x \geqq 1$ ならば文30へ進む。

**論理 IF 文** 基本形は

IF（算術式 等号または不等号 算術式） その場合に実行すべき文

で，その場合の等号，不等号は次のように表す。

| 数学記号 | < | ≦ | ≧ | > | = | ≠ |
|---|---|---|---|---|---|---|
| FORTRAN | .LT. | .LE. | .GE. | .GT. | .EQ. | .NE. |

なお，一般には（ ）内に，

論理変数

論理式

等式や不等式を .AND. や .ŌR. で結んだもの

を書くことができる。

〔注意〕 比較のための等号や不等の左辺と右辺の型は一致していなければならない。たとえば，IF（X－1） …は誤り。

## A.10 DO文

これはBASICのFOR文に相当するもので，次の形式で書く。

DŌ　端末文番号　制御変数名＝始値，終値，増分

ここで端末文番号というのは，くりかえし部分の最後の文（これを端末文という）の文番号である。たとえば，

```
      |DŌ  30  I=1,7,2        ↑
      |  …                    |
      |  …                    | この部分が反復される
   30 |A=B+C                  ↓
```

端末文は，上記のように代入文その他であってもよいが，BASICのNEXT文のように，端末文を独立させておく方が明快なので，

|端末文番号| |CŌNTINUE

という文を用いるのがよいとされている。

〔注意〕 制御変数名は整数型に限る。始値，終値，増分も整数型で，そこには式を書けない。また，0や負の始値，負の増分は許されない。

（例）
```
|DŌ  100  K=0,10      ┐
|DŌ  200  L=10,1,-1   │ いずれもエラーになる。
|DŌ  300  M=1,N+1     ┘
```

もう一つ重要な注意は，指定された反復を全部終って端末文の次の文に進んだとき，制御変数の値は不定になる（でたらめの値になるかもしれない）ということである。たとえば

```
      |DŌ  50  I=1,10
      |  …
   50 |CŌNTINUE
```

の場合，反復終了後は I＝10 になると思っている人がよくいるが，それは誤りで，さりとて I＝11 になるわけでもなく，仮りにその規則を調べても，他機種の扱いはまた違っているかもしれない。

## A.11 組み込み関数†

FORTRAN で使用できる組み込み関数†は，機種によって多少異なるが，規格（JIS の水準7000）では次のようになっている††。

| 機能 | 実数型 | 倍精度実数型 | 整数型 |
|---|---|---|---|
| $e^x$ | EXP(X) | DEXP | — |
| $\log_e x$ | ALŌG(X) | DLŌG | — |
| $\log_{10} x$ | ALŌG10(X) | DLŌG10 | — |
| $\sin x$ | SIN(X) | DSIN | — |
| $\cos x$ | CŌS(X) | DCŌS | — |
| $\tanh x$ | TANH(X) | — | — |
| $\sqrt{x}$ | SQRT(X) | DSQRT | — |
| $\arctan x$ | ATAN(X) | DATAN | — |
| $\arctan(x/y)$ | ATAN2(X,Y) | — | — |
| $x$を$y$で割った余り | AMŌD(X,Y) | DMŌD | MŌD |
| $\|x\|$ | ABS(X) | DABS | IABS |
| 小数点以下切り捨て | AINT(X) | DINT | INT |
| $\max(i_1, i_2, \cdots)$ | AMAX0(I1,I2,…) | — | MAX0 |
| $\max(x_1, x_2, \cdots)$ | AMAX1(X1,X2,…) | DMAX1 | MAX1 |
| $\min(i_1, i_2, \cdots)$ | AMIN0(I1,I2,…) | — | MIN0 |
| $\min(x_1, x_2, \cdots)$ | AMIN1(X1,X2,…) | DMIN1 | MIN1 |
| 符号††† | SIGN(X,Y) | DSIGN | ISIGN |
| $x-\min(x, y)$ | DIM(X,Y) | | IDIM(X,Y) |

| | | | |
|---|---|---|---|
| 実数型を整数型に変換 | IFIX | 実数部 | REAL |
| 整数型を実数型に変換 | FLŌAT | 虚数部 | AIMAG |
| 単精度化 | SNGL | 共役複素数 | CŌNJG |
| 倍精度化 | DBLE | 複素数化 | CMPLX(X,Y) |

† 文法的には「組み込み関数」と「基本外部関数」に分かれるが，普通の利用者にとっては同じこと。要するに「処理系が用意してくれた関数」。

†† このほかに複素数型のCEXP，CLŌG，CSIN，CCŌS，CSQRT，CABSがある。

††† $x$ の絶対値に $y$ の符号をつけたもの。

## A.12 サブルーチン

サブルーチンの書きかた，使いかたは，BASICの場合と著しく異なる。すなわち，FORTRANのサブルーチンは

独立したプログラムである（「プログラム単位」という）

——したがって，変数名や文番号は，呼び出し側のプログラムや他のサブルーチンと独立であり，単独にコンパイルできる。

呼び出しの際の識別は「サブルーチン名」による。

データの受け渡しは，すべて「引数」によって行なう。

という文法になっている。そのため，大規模なプログラムを何人かで分けて作る場合に一つ一つ独立に作ることができ，また一つのサブルーチンをいろいろなプログラムに組み込むにも便利である。

**サブルーチンの書きかた**　サブルーチンは次の形式で書く。

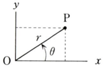
```
SUBRŌUTINE サブルーチン名(引数の列)
   処理の内容を記述
   するプログラム
END
```

サブルーチン名は6字以内の英数字で先頭は英字とする。「引数の列」とは，変数名や配列名をコンマで区切って並べたもの。入力用と出力用の引数の区別はなく，入出力両用（すなわち，値を変更して返す）も許される。

出口にはRETURNと書く（これはBASICと同じ）。

（例）　極座標$r$，$\theta$の値を直交座標$x$，$y$に変換するサブルーチンを作ってみよう。サブルーチン名はHENKANとし，$r$，$\theta$，$x$，$y$をR，T，X，Yで表すことにすれば，次のようになる。

```
SUBRŌUINE HENKAN(R,T,X,Y)
X=R*CŌS(T)
Y=R*SIN(T)
RETURN
END
```



**サブルーチンの呼び出しかた** サブルーチンを呼び出すには CALL 文を用いる。これは次の形式で書く。

CALL サブルーチン名（引数の列）

（例） いま作ったサブルーチン HENKAN を呼び出して，$r=3$，$\theta=0.1$（ラジアン）を $x$，$y$ に変換するには

```
CALL HENKAN(3.0,0.1,X,Y)
```

とすればよい。このように，CALL 文の引数でサブルーチンに値を引き渡すだけのものには，定数や式を書いてもよい。

サブルーチンと呼び出し側のプログラムとでは変数名が独立であるから，引数の名前が一致する必要ない。引数は名前の一致によって結合されるのではなく，順序に従って結合される。

（例）
```
CALL HENKAN(RR,THETA,X1,X2)

SUBROŪTINE HENKAN(R,T,X,Y)
```

**整合寸法** 配列を引数として受け渡す場合，呼び出し側のプログラムで宣言した寸法と，呼び出された側のプログラムで宣言した寸法は一致させておく必要がある。そこで，呼び出し側で宣言した寸法を引数として受け取って，それに合わせて配列宣言を行なうことができるようになっている。

（例）
```
  SUBROŪTINE CLEAR(A,M,N)
  DIMENSIŌN A(M,N)
  DŌ 1 I=1,M
  DŌ 1 J=1,M
1 A(I,J)=0.0
  RETURN
  END
```

ただし，このような書きかたが許されるのは，配列名と寸法がともに引数の場合に限る。また，寸法として式は書けない。

（誤りの例）
```
SUBROŪTINE S(X,Y,N)
DIMENSIŌN X(N+1),A(N,N)
```

## A.13 入 出 力

FORTRANの入出力の書きかたは，かなりめんどうである。それは，常に書式を指定しなければならないからである。拡張BASICに

```
PRINT USING "書式"；変数名の列
```

という書きかたがあるが，それと同様なことを常に（しかも，もっと難解な規則に従って）書かなければならない†。

**READ文**　FORTRANにはINPUTという文はなくて，いつでもREAD文を用いる。書きかたは，

```
READ(装置番号，書式の文番号) 変数名の列
```

である。装置番号は，その名のとおり装置を指示する番号であるが，普通，データを読み込むための標準的な入力装置を5番にする。書式の文番号というのは，あとで述べるFORMAT文の文番号であり，右に列挙された変数の値を，そのFORMAT文の書式に従って読み込むことになる。

**WRITE文**　出力には，普通，WRITE文を用いる。書きかたは

```
WRITE(装置番号，書式の文番号) 変数名の列
```

である。標準的出力の装置番号は，普通，6である。

**配列の入出力**　配列の要素全部を入出力するのであれば，「変数名の列」に配列名を書けばよい（カッコや添字は付けない）。ただし行列の場合，列番号順（たとえば $a_{11}$，$a_{21}$，$a_{31}$，$a_{12}$，$a_{22}$，$a_{32}$，$a_{13}$，…）になるので，行列を扱う場合には転置行列のような感じになる。なお，次のような「DO型並び」という書きかたもできる。

（例）
```
WRITE(6,80) (X(I),I=11,18)
READ(5,2) ((A(I,J),J=1,3),I=1
     (前行の続き) ,7)
```

† ただしFORTRAN 77や一部の拡張FORTRANでは

```
READ(装置番号，*) 変数名の列
WRITE(装置番号，*) 変数名の列
```

という書きかたができる。

**FORMAT 文**　これは書式を指定するもので，次の形式で書く。

FŌRMAT (欄記述子の列)

ここで「欄記述子の列」というのは，欄記述子をコンマで区切って並べたもの。「欄記述子」というのは，簡単にいえば「一つの変数の書式を記述したもの」またはそのくりかえし，または「出力すべき文字列」または「空白を入れる指定」である。よく使われる欄記述子には次のようなものがある。

F$m$.$n$　欄の長さが$m$桁で，その右から$n$桁を小数部にすることを表す。実数型の入出力に用いる。

E$m$.$n$　欄の長さが$m$桁で，その右4桁を「E±××」の形の指数部にあて，その前の$n$桁を小数部にする。実数型の入出力に用いる。

D$m$.$n$　これはE$m$.$n$とだいたい同じ。倍精度実数型用。

I$m$　欄の長さが$m$桁で，それがすべて整数部にあてられることを表す（欄のいちばん右が1の位）。整数型の入出力に用いる。

L$m$　欄の長さが$m$桁で，そこに論理値が「真」をT,「偽」をFで表して記録されることを表す。

A$m$　欄の長さが$m$桁で，それが$m$字の文字列として，文字型変数に対応することを示す。

$n$X　入力の場合は，$n$桁を読みとばすことを表し，出力の場合は$n$字ぶんの空白を入れることを表す。

複素数型の入出力は実数型2箇（実数部，虚数部の順）として扱う。

**行送りの制御**　出力の場合，行の第1字目にあたる文字は印字されず，行送りの制御文字として扱われる。すなわち，これが

空白ならば普通の改行（シングル・スペース）

0ならばもう一度改行（ダブル・スペース）

1ならば次のページに移る（改ページ）

＋ならば改行せず復帰だけ（重ね打ち）

## A.14 FORTRANプログラムの例

1) 2.2.1節のプログラムのFORTRAN訳

```
      READ(5,1) A,B,H
    1 FŌRMAT(3F10.5)              書式指定が必要
      S=(A+B)*H/2.0               小数点を忘れるな！
      WRITE(6,2) S
    2 FŌRMAT('␣',F10.5)
      STŌP
      END
```

2) 2.3.1節のプログラムのFORTRAN訳

```
      DŌ 30 N=1,10
      EN=N                        実数型に変換
      R=1.0/EN                    小数点を忘れるな！
      S=SQRT(EN)                  関数名の相違に注意
      N2=N*N
      WRITE(6,1) N,N2,R,S
    1 FŌRMAT('␣',2I5,2F10.5)
   30 CŌNTINUE
      STŌP
      END
```

3) 2.4.1節のプログラムのFORTRAN訳

```
      DIMENSIŌN X(100)
      READ(5,1) N
    1 FŌRMAT(I5)
      READ(5,2) (X(I),I=1,N)
    2 FŌRMAT(F10.5)
      W=0.0
      DŌ 10 I=1,N
         W=W+X(I)
   10 CŌNTINUE
      H=W/FLŌAT(N)                型変換が必要
```

```
      W=0.0
      DŌ 20 I=1,N
        W=W+(X(I)-H)**2            べき乗は**印
   20 CŌNTINUE
      S=SQRT(W/FLŌAT(N))
      WRITE(6,3) H,S
    3 FŌRMAT('␣HEIKIN=',F10.5,
     1 '␣HYŌUZYUNHENSA',F10.5)
      STŌP
      END
```

4) 2.5.1 節のプログラムの FORTRAN 訳

```
      INTEGER B,K,S,E,G
      DIMENSIŌN B(100),K(100),
     1          S(100),E(100)
      READ(5,1)N
    1 FŌRMAT(I5)
      DŌ 60 I=1,N
      READ(5,5) B(I),K(I),S(I),E(I)
    5 FŌRMAT(4I5)
   60 CŌNTINUE
      WRITE(6,2)
    2 FŌRMAT('␣BANGŌU,TENSUU,HANTEI')
      DŌ 100 I=1,N
      G=K(I)+S(I)+E(I)
      IF(G.LT.200) GŌ TŌ 130
      WRITE(6,6) B(I),G
    6 FŌRMAT('␣',2I5,'Ō')
      GŌ TŌ 140
  130 WRITE(6,7) B(I),G
    7 FŌRMAT('␣',2I5,'X')
  140 CŌNTINUE
      END
```

## A.15 そ の 他

**DATA文** FORTRANにもDATA文があるが，それは「入力データを定義したもの」ではなくて，「変数の初期値を与えるもの」である。たとえば，

```
DATA A/1.0/
```

と書けば，プログラムの実行開始時点における変数Aの値は1になる。一般的な書きかたは

DATA 変数名の列/左記に対応する定数の列/

またはそれをコンマで区切って並べたものである。

（例）

```
DATA I,J,K/8,1,5/
DATA L/6/,M/3/,N1,N2/3,5/
```

「変数名の列」の所に配列名を書いてよい。ただし第1添字，第2添字，第3添字の順に変るので行列の扱いには注意を要する。たとえば

```
DIMENSIŌN A(2,2)
DATA A/1.0,2.0,3.0,4.0/
```

とすると次のようになる。

$$a_{11}=1 \qquad a_{12}=3$$
$$a_{21}=2 \qquad a_{22}=4$$

**STOP文** FORTRANのSTOP文は「ここで，このプログラムの実行を終了する」という意味になり，普通はすぐ次の人の仕事に移る。単なる一時停止は

```
PAUSE
```

と書く。これだと，その次の文から実行を再開できる。

**END行** これはプログラム単位（すなわち，メインプログラム，サブルーチンなど）の区切りを表す。BASICの場合は上記のSTOP文のような意味で何箇所にでも書けたが，FORTRANの場合はプログラム単位の末尾にしか書けないので注意を要する。

# 解　　答

## 2.2.1 演習

1

(1) －3＊C　(2) （A＋B）＊C　(3) A＊B

2

(1) 1／（X＋Y）　(2) 1／（X＊Y）
(3) Y／（2＊X）　(4) B／A＋D／C
(5) （A＋B）／（A1＋B1）

## 2.2.2 演習

1

(1) SQR（1－X^2）
(2) LŌG（LŌG（X））
(3) EXP（2＊X）
(4) SIN（3.1416／180＊10）
(5) ABS（X）－ATN（Y）

2

```
10 INPUT N,A
20 LET S=(A/2)^2*N/TAN(3.14
   16/N)
30 PRINT S
40 END
```

## 2.2.3 応 用 問 題

1

```
10 INPUT "A,B=";A,B
20 LET A1=A+B
30 LET A2=A-B
40 LET A3=A*B
50 LET A4=A/B
60 PRINT "WA,SA,SEKI,SYŌ"
70 PRINT A1;A2;A3;A4
80 END
```

2

i)

```
10 INPUT "SESI=";C
20 LET F=9*C/5+32
30 PRINT "F=";F
40 END
```

```
10 INPUT "KASI=";F
20 LET C=5/9*(F-32)
30 PRINT "C=";C
40 END
```

ii)

```
10 INPUT "MILLI=";M
20 LET I=M/25.4
30 PRINT "INCH=";I
40 END
```

```
10 INPUT "INCH=";I
20 LET M=25.4*I
30 PRINT "MILLI=";M
40 END
```

iii)

```
10 INPUT "PŌUND=";P
20 LET K=0.45359*P
30 PRINT "KILŌ_GRAM=";K
40 END
```

```
10 INPUT "KG=";K
20 LET P=K/0.45359
30 PRINT "PŌUND=";P
60 END
```

iv)

```
10 INPUT "WATT=";W
20 LET B=W/736
30 PRINT "BARIKI=";B
40 END
```

```
10 INPUT "BARIKI=";B
20 LET W=736*B
30 PRINT "WATT=";W
40 END
```

v)

```
10 INPUT "KIATU=";K
```

```
20 LET M=1013.25*K
30 LET S=760*K
40 PRINT "MIRIBAR=";M,"SU
   IGIN=";S
50 END

10 INPUT "SUIGINTYU=";S
20 LET M=(1013.25/760)*S
30 LET K=S/760
40 PRINT "MILLIBAR=";M,"KIA
   TU=";K
50 END
```

vi)

```
10 INPUT "BYŌSŌKU=";B
20 LET J=3.6*B
30 PRINT "JISŌKU(KM)=";J
40 END

10 INPUT "JISŌKU=";J
20 LET B=J/3.6
30 PRINT "BYŌSŌKU=";B
40 END
```

vii)

```
19 INPUT "RADIAN=";R
20 LET D=(180/3.1416)*R
30 PRINT "DŌ=";D
40 END

10 INPUT "DŌ=";D
20 LET R=(3.1416/180)*D
30 PRINT "RADIAN=";R
40 END
```

**3**

i)

```
10 INPUT T,M,H
20 LET X=(T-M)/H*10+50
30 PRINT "HENSATI=";X
40 END
```

ii)

```
10 INPUT T1,T2
20 LET X=0.99*T1+0.36*T2+41
   .5
30 PRINT "HUKAI SISU="; X
40 END
```

iii)

```
10 INPUT T,K
20 LET X=20*LŌG(T/K)/LŌG(10
   )
30 PRINT "DB=";X
40 END
```

**4**

i)

```
10 INPUT A,H
20 LET X=(A*H)/2
30 PRINT "MENSEKI=";X
40 END
```

ii)

```
10 INPUT "HANKEI=";R
20 LET X=3.1416*R*R
30 PRINT "EN NŌ MENSEKI=";X
40 END
```

iii)

```
10 INPUT "HANKEI=";R
20 LET V=(4/3)*3.1416*R*R*R
30 LET S=4*3.1416* R*R
40 PRINT "KYU NŌ TAISEKI=";
   V,"HYŌMENSEKI=";S
50 END
```

iv)

```
10 INPUT R,H
20 LET V=(3.1416/3)*R*R*H
30 PRINT "ENSUI NŌ TAISEKI=
   ";V
40 END
```

v)

```
10 INPUT "TŌRUS NŌ HANKEI R
   1,R2=";R1,R2
20 LET S=4*3.1416*3.1416*R1
   *R2
30 PRINT "TŌRUS NŌ HYŌMENS
   EKI=";S
40 END
```

vi)

```
10 INPUT "SYUJIKU NŌ NAGASA
   A,B,C NŌ 1/2=";A,B,C
20 LET V=(4/3)*3.1416*A*B*C
30 PRINT "DAENTAI NŌ TAIS
   EKI=";V
40 END
```

**5**

```
10 PRINT SQR(2);"___1.41421
   3562373"
20 PRINT LŌG(2);"___0.69314
   7180560"
```

```
30 PRINT EXP(1);"___2.71828
   1828459"
40 PRINT SIN(3.1416/18);"__
   _0.173648177667"
50 PRINT ATN(1);"___0.78539
   8163397"
```

6

```
10 INPUT A,B
20 LET S=SIN(A+B)
30 LET U=SIN(A)*CŌS(B)+CŌS(
   A)*SIN(B)
40 PRINT "SIN(A+B)=___SAHEN
   =";S,"UHEN=";U,"SA=";ABS
   (S-U)
50 LET S=CŌS(A+B)
60 LET U=CŌS(A)*CŌS(B)-SIN(
   A)*SIN(B)
70 PRINT "CŌS(A+B)=___SAHEN
   =";S,"UHEN=";U,"SA=";ABS
   (S-U)
80 LET T1=TAN(A)
90 LET T2=TAN(B)
100 LET S=TAN(A+B)
110 LET U=(T1+T2)/(1-T1*T2)
120 PRINT "TAN(A+B)=___SAHEN
    =";S,"UHEN=";U,"SA=";ABS
    (S-U)
130 END
```

## 2.3.1 演習

1

```
10 FŌR X=1 TŌ 2.01 STEP 0.1
20 LET Y=2*X*X+3*X-1
30 PRINT X,Y
40 NEXT X
50 END
```

2

```
10 INPUT A,B,C,X1,X2,D
15 PRINT "X=","Y="
20 FŌR X=X1 TŌ X2 STEP D
30 LET Y=A*X*X+B*X+C
40 PRINT X,Y
50 NEXT X
60 END
```

## 2.3.2 演習

1

```
10 LET A=1
20 LET S=1
30 FŌR K=1 TŌ 10
40 LET A=A/K
50 LET S=S+A
60 PRINT K,A,S
70 NEXT K
80 END
```

2

```
1 INPUT X
10 LET A=1
20 LET S=1
30 FŌR K=1 TŌ 10
40 LET A=A*X/K
50 LET S=S+A
60 NEXT K
70 PRINT S
80 END
```

3

(1)

```
10 INPUT N,X
20 LET A=1
30 LET S=1
40 FŌR K=1 TŌ N
50 LET A=-A*X*X/(2*K*(2*K-1
   ))
60 LET S=S+A
70 NEXT K
80 PRINT "N=";N,"X=";X,"WA=
   ";S
90 END
```

(2)

```
10 INPUT N,X
20 LET B=X
30 LET S=X
40 FŌR K=1 TŌ N-1
50 LET B=-X*X*B
60 LET A=B/(2*K+1)
70 LET S=S+A
80 NEXT K
90 PRINT "N=";N,"X=";X,"WA=
   ";S
100 END
```

(3)

```
10 INPUT N,X
20 LET S=0
30 FŌR K=1 TŌ N STEP 2
40 LET A=SIN(K*X)/K
50 LET B=SIN((K+1)*X)/(K+1)
60 LET S=S+A-B
70 NEXT K
80 PRINT S
90 END
```

## 2.3.3 応 用 問 題

**1**

i)

```
10 PRINT "X","SIN(X)","CŌS(
   X)","TAN(X)"
20 LET A=3.1416/180
30 FŌR X=0 TŌ 90 STEP 10
40 LET Y=A*X
50 PRINT X,SIN(Y),CŌS(Y),TA
   N(Y)
60 NEXT X
70 END
```

ii)

```
10 PRINT "X","LŌG10(X)"
20 LET A=1/LŌG(10)
30 FŌR X=0.1 TŌ 2.01 STEP 0
   .1
40 PRINT X, A*LŌG(X)
50 NEXT X
60 END
```

iii)

```
10 PRINT "N","N!"
20 PRINT " 1"," 1"
30 LET N1=1
40 FŌR N=2 TŌ 10
50 LET N1=N1*N
60 PRINT N,N1
70 NEXT N
80 END
```

**2**

```
10 PRINT "LITER","G1","G2",
   "G3","G4"
20 FŌR X=1 TŌ 30.01 STEP 0.
   1
30 LET Y1=135*X
40 LET Y2=138*X
50 LET Y3=145*X
60 LET Y4=155*X
80 PRINT X,Y1,Y2,Y3,Y4
90 NEXT X
100 END
```

**3**

```
10 PRINT "GANKIN 10000YEN,X
   NEN GŌ NŌ GANRI GŌUKEI(N
   EN HUKURI)"
20 PRINT X;"(NEN)","3.5","5
   ","6";"(PERCENT)_YEN"
30 FŌR X=1 TŌ 20
40 LET Y1=10000*1.035^X
50 LET Y2=10000*1.05^X
60 LET Y3=10000*1.06^X
70 PRINT X,Y1,Y2,Y3
80 NEXT X
90 END
```

**4**

```
10 PRINT "TUMITATE YŌKIN 10
   000YEN/NEN NŌ GANRI GŌU
   KEI"
20 PRINT "X_NEN", "3.5","5"
   ,"6";"(PERCENT)_YEN"
30 LET Y1=10000
40 LET Y2=10000
50 LET Y3=10000
60 FŌR X=1 TŌ 20
70   LET Y1=Y1*1.035
80   LET Y2=Y2*1.05
90   LET Y3=Y3*1.06
100  PRINT X,Y1,Y2,Y3
110  LET Y1=Y1+10000
120  LET Y2=Y2+10000
130  LET Y3=Y3+10000
140 NEXT X
150 END
```

**5**

```
10 PRINT "TAISYŌKUKIN NŌ ZA
   NDAKA"
20 PRINT "X_NEN GŌ","3.5","
   5","6";(PERCENT)_YEN"
30 LET Y1=10000000
40 LET Y2=10000000
50 LET Y3=10000000
60 FŌR X=1 TŌ 10
70   LET Y1=Y1*1.035
80   LET Y2=Y2*1.05
90   LET Y3=Y3*1.06
100  LET Y1=Y1-1000000
110  LET Y2=Y2-1000000
120  LET Y3=Y3-1000000
130  PRINT X,Y1,Y2,Y3
140 NEXT X
150 END
```

**6**

i)

```
10 PRINT "X(N)^X(N)___(LIM
   X TŌ 0),(X(N)=10^(-N))"
20 FŌR N=1 TŌ 10
30   LET Y=(10^(-N))^(10^(-
     N))
```

```
40    PRINT N,Y
50 NEXT N
60 END
```

ii)

```
10 PRINT "SIN(X)/X___(LIM X TŌ 0),(X(N)=10^(-N))"
20 FŌR N=1 TŌ 10
30    PRINT N,SIN(10^(-N))/(10^(-N))
40 NEXT N
50 END
```

iii)

```
10 PRINT "(1+1/N)^N___(LIM N TŌ LARGE)"
20 FŌR M=1 TŌ 4
25    LET N=10^M
30    PRINT N,(1+1/N)^N
40 NEXT M
50 END
```

iv)

```
10 PRINT "R=1,NŌ NAISETU(GAISETU)TAKAKUKEI NŌ SYU"
20 FŌR N=3 TŌ 10
30    LET T=3.1416/N
40    PRINT N,2*N*SIN(T),2*N*TAN(T)
50 NEXT N
60 END
```

v)

```
10 PRINT "TAN(X) NŌ RENBUNSU KINZI"
20 INPUT "X=";X
30 LET B=11
40 FŌR N=5 TŌ 1 STEP -1
50    B=2*N-1-X*X/B
70 NEXT N
80 LET Y=X/B
90 PRINT "X=";X,"TAN(X)=";Y
100 END
```

### 2.4.3 応 用 問 題

1

```
10 DIM A(3),B(3),X(3)
20 INPUT A(1),A(2),A(3)
30 INPUT B(1),B(2),B(3)
40 PRINT
50 FŌR I=1 TŌ 3
60      LET X(I)=A(I)+B(I)
70 NEXT I
80 PRINT "VECTŌR NŌ WA=";X(1),X(2),X(3)
90 FŌR I=1 TŌ 3
100     LET X(I)=A(I)-B(I)
110 NEXT I
120 PRINT "VECTŌR NŌ SA=";X(1),X(2),X(3)
130 LET Y=0
140 FŌR I=1 TŌ 3
150    LET Y=Y+A(I)*B(I)
160 NEXT I
170 PRINT "VECTŌR NŌ NAISEKI=";Y
180 LET X(1)=A(2)*B(3)-A(3)*B(2)
190 LET X(2)=A(3)*B(1)-A(1)*B(3)
200 LET X(3)=A(1)*B(2)-A(2)*B(1)
210 PRINT "VECTŌR NŌ GAISEKI="; X(1),X(2),X(3)
220 LET Y=SQR(A(1)*A(1)+A(2)*A(2)+A(3)*A(3))
230 PRINT "VECTŌR A NŌ NAGASA=";Y
240 END
```

2

```
10 DIM A(3,3)
20 FŌR I=1 TŌ 2
30      FŌR J=1 TŌ 2
40          READ A(I,J)
50      NEXT J
60 NEXT I
70 LET D=A(1,1)*A(2,2)-A(1,2)*A(2,1)
80 PRINT "DET(A)="; D
90 FŌR I=1 TŌ 3
100     FŌR J=1 TŌ 3
110         READ A(I,J)
120     NEXT J
130 NEXT I
140 LET D=A(1,1)*A(2,2)*A(3,3)+A(1,2)*A(2,3)*A(3,1)+A(1,3)*A(2,1)*A(3,2)-A(1,3)*A(2,2)*A(3,1)-A(1,1)*A(2,3)*A(3,2)-A(2,1)*A(1,2)*A(3,3)
150 PRINT "DET(A)=";D
160 END
```

**3**

```
1 DIM A(100)
10 READ N,X
20 FŌR I=0 TŌ N
30     READ A(I)
40 NEXT I
50 LET Y=A(0)
60 FŌR I=1 TŌ N
70 LET Y=Y*X+A(I)
80 NEXT I
90 PRINT "N_JI TAKŌUSIKI NŌ
   ATAI="; Y
100 END
```

**4**

```
10 DIM A(21)
20 LET A(0)=0
30 LET A(1)=1
40 FŌR I=1 TŌ 20
50     A(I+1)=A(I)+A(I-1)
60 NEXT I
70 FŌR I=1 TŌ 16 STEP 5
80     PRINT A(I-1),A(I),A(I
   +1),A(I+2),A(I+3)
90 NEXT I
100 PRINT A(20)
110 END
```

**5**

```
10 DIM X(11),Y(11)
20 LET N=1
30 LET X(0)=1
40 LET X(1)=1
45     LET Y(0)=1
50 PRINT "2_KŌU KEISU"
60 PRINT "N,C(0),C(1),---,C
   (10)"
70 PRINT N;X(0);X(1)
80 FŌR N=2 TŌ 10
90 FŌR I=1 TŌ N-1
100 LET Y(I)=X(I-1)+X(I)
110 NEXT I
120 LET Y(N)=1
130 FŌR I=1 TŌ N
140 LET X(I)=Y(I)
150 NEXT I
160 PRINT N;
170 FŌR J=0 TŌ N
180 PRINT X(J);
190 NEXT J
195 PRINT
200 NEXT N
210 END
```

**6**

```
10 DIM A(13,13),L(13)
20 FŌR I=1 TŌ 13
30 READ L(I)
40 NEXT I
50 PRINT TAB(2);
60 FŌR I=1 TŌ 13
70 PRINT TAB(5*I);I;
80 NEXT I
85 PRINT
90 FŌR I=1 TŌ 13
100 PRINT I;
110 FŌR J=I TŌ 13
120 LET A(I,J)=L(J)-L(I)
130 PRINT TAB(5*J);A(I,J);
140 NEXT J
145 PRINT
150 NEXT I
160 END
200 DATA 0,29,84,105,121,180
   ,257
210 DATA 294,366,396,446,514
   ,553
```

結果は次のようになる。

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 29 | 84 | 105 | 121 | 180 | 257 | 294 | 366 | 396 | 446 | 514 | 553 |
| 2 | | 0 | 55 | 76 | 92 | 151 | 228 | 265 | 337 | 367 | 417 | 485 | 524 |
| 3 | | | 0 | 21 | 37 | 96 | 173 | 210 | 282 | 312 | 362 | 430 | 469 |
| 4 | | | | 0 | 16 | 75 | 152 | 189 | 261 | 291 | 341 | 409 | 448 |
| 5 | | | | | 0 | 59 | 136 | 173 | 245 | 275 | 325 | 393 | 432 |
| 6 | | | | | | 0 | 77 | 114 | 186 | 216 | 266 | 334 | 373 |
| 7 | | | | | | | 0 | 37 | 109 | 139 | 189 | 257 | 296 |
| 8 | | | | | | | | 0 | 72 | 102 | 152 | 220 | 259 |
| 9 | | | | | | | | | 0 | 30 | 80 | 148 | 187 |
| 10 | | | | | | | | | | 0 | 50 | 118 | 157 |
| 11 | | | | | | | | | | | 0 | 68 | 107 |
| 12 | | | | | | | | | | | | 0 | 39 |
| 13 | | | | | | | | | | | | | 0 |

**7**

(略)

## 2.5.1 演習

**1**

```
10 FŌR I=1 TŌ 10
20 INPUT B,K,S,E
30 LET G=K+S+E
40 IF G>=250 THEN 80
50 IF G<250 THEN 100
60 PRINT "B"
70 GŌTŌ 110
80 PRINT "A"
90 GŌTŌ 110
100 PRINT "X"
110 NEXT I
120 END
```

2

```
10 INPUT X
20 IF X<2 THEN 50
30 LET Y=10
40 GŌTŌ 60
50 LET Y=2.5*X
60 PRINT Y
70 END
```

3

```
10 INPUT N
20 LET M=N/2
30 IF M=INT(M) THEN 60
40 PRINT "K"
50 GŌTŌ 70
60 PRINT "G"
70 END
```

## 2.5.2 演習

1

```
10 INPUT "A,B,C=",A,B,C
11 IF A=0 THEN 162
20
30     上記と同じ
 ⋮
160
161 GŌTŌ 164
162 LET X=-C/B
163 PRINT "X=";X
164 END
```

2

```
10 INPUT "A,B,C=",A,B,C
20 D=B*B-4*A*C
30 S=-B/(2*A)
40 T=SQR(ABS(D))/(2*A)
50 IF D<0 THEN 200
60 IF D=0 THEN 170
70 REM D>0 NŌ BAAI
80 IF S<0 THEN 110
90 X1=S+T
100 GŌTŌ 120
110 X1=S-T
120 X2=(C/A)/X1
130 PRINT "X1=";X1
140 PRINT "X2=";X2
150 GŌTŌ 220
160 REM D=0 NŌ BAAI
170 PRINT "X=";S
180 GŌTŌ 220
190 REM D<0 NŌ BAAI
200 PRINT "X1=";S;"+";T;"I"
210 PRINT "X2=";S;"-";T;"I"
220 END
230 RUN
```

## 2.5.3 応 用 問 題

1

```
10 INPUT "SEIREKI(NEN)N=";N
20 IF INT(N/4)<>N/4 THEN 70
30 IF INT(N/100)=N/100 THEN
   60
40 PRINT "URUU DŌSI"
50 GŌTŌ 70
60 IF INT(N/400)=N/400 THEN
   40
70 END
```

2

```
10 INPUT "Y-NEN,M-GATU,D-NI
   TI=";Y,M,D
20 IF M<4 THEN 70
30 IF M>4 THEN 50
40 IF D=1 THEN 70
50 PRINT Y;M;D;"-ŌSŌUMARE"
60 STŌP
70 PRINT Y;M;D;"-HAYAUMARE"
80 END
```

3

```
10 INPUT "HŌNSU N=";N
20 IF N<10 THEN 80
30 LET I=INT(N/10)
40 IF I=N/10 THEN 100
50 LET X=3000*I+330*(N-I*10
   )
60 PRINT N;"-HŌN",X;"YEN"
70 GŌTŌ 200
80 LET X=330*N
90 GŌTŌ 60
100 LET X=3000*I
110 GŌTŌ 60
200 END
```

4

```
10 INPUT "-GATU,-NITI=",M,D
20 IF M<1  THEN 500
30 IF M>12  THEN 500
40 IF M=1  THEN 100
50 IF M=2  THEN 250
60 IF M>7  THEN 200
70 IF INT(M/2)=M/2 THEN 150
100 IF D<1  THEN 500
```

```
110 IF D>31    THEN 500
120 GŌTŌ    600
150 IF D<1    THEN 500
160 IF D>30    THEN 500
170 GŌTŌ  600
200 IF INT(M/2)=M/2 THEN 100
210 GŌTŌ 150
250 IF D<1 THEN 500
260 IF D<=29 THEN 600
270 GŌTŌ 500
500 PRINT M;"GATU";D;"NITI"
    WA ARIENAI"
510 GŌTŌ 610
600 PRINT "ŌK" M,D
610 END
```

**5**

最高100枚までとする

```
10 DIM A(100)
20 INPUT N,L,M
22 LET M1=1
24 FŌR I=1 TŌ M
26 LET M1=M1*10
28 NEXT I
30 FŌR I=1 TŌ N
40 READ  A(I)
50 NEXT I
60 FŌR I=1 TŌ N
70 IF A(I)-M1*INT(A(I)/M1)=
   L THEN 90
80 GŌTŌ 100
90 PRINT "ATARI";^A(I)
100 NEXT I
110 END
```

**6**

```
10 INPUT X,E
20 LET A1=X
30 LET B1=X/A1
40 LET A2=(A1+B1)/2
50 IF ABS(A1-B1)<E   THEN 80
60 LET A1=A2
70 GŌTŌ 30
80 PRINT "SQR(X)=";A2
90 END
```

**7**

i)

```
10 LET I=0
20 LET I=I+1
```

| 反復手続き |
|---|

```
100 IF I>=N THEN 120
110 GŌTŌ 20
120 END
```

ii)

```
10 LET X=A
```

| 反復手続き |
|---|

```
100 IF C<0   THEN   200
110 IF X>=B   THEN   300
120 LET X=X+C
130 GŌTŌ 20
200 IF X<=B   THEN 300
210 LET X=X+C
220 GŌTŌ 20
300 END
```

iii)

```
10 PRINT "X","EXP(X)"
20 LET X=0
30 LET Y=EXP(X)
40 PRINT X,Y
50 IF X>=1 THEN 80
60 LET X=X+0.1
70 GŌTŌ 30
80 END
```

## 2.6.3 演習

**1**

```
(1) DEF FNS(X)
        =(EXP(X)-EXP(-X))/2
(2) DEF FNC(X)
        =(EXP(X)+EXP(-X))/2
(3) DEF FNT(X)
        =FNS(X)/FNC(X)
(4) DEF FNA(X)
        =ATN(X/SQR(1-X*X))
(5) DEF FNB(X)
        =3.1416/2-FNA(X)
```

## 2.6.6 応用問題

**1**

```
10 INPUT N¥
20 GŌSUB 200
30 GŌSUB 200
40 PRINT "HAPPY BIRTHDAY
   DEAR ";N¥;" SAN"
50 GŌSUB 200
60 STŌP
200 PRINT "HAPPY BIRTHDAY TŌ
    YŌU"
210 RETURN
```

```
220 END
```

〔別解〕

```
10 A¥="HAPPY BIRTHDAY"
20 B¥="TŌ YŌU"
30 INPUT N¥
40 GŌSUB 110
50 PRINT A¥;B¥
60 PRINT A¥;B¥
70 PRINT A¥;"DEAR ";N¥;" SAN"
80 PRINT A¥;B¥
90 GŌSUB 110
100 GŌTŌ 30
110 FŌR I=1 TŌ 5
120 PRINT
130 NEXT I
140 RETURN
150 END
```

**2**

```
10 DATA "=====   VVVV "
20 DATA "I   I     II   "
30 DATA "I___I_II_II_ "
40 DATA "I           I "
50 DATA "I___ŌŌŌ_ŌŌŌI_"
60 DATA "    ŌŌŌ ŌŌŌ   "
100 DIM  D¥(6)
110 FŌR I=1 TŌ 6
120    READ D¥(I)
130    PRINT D¥(I)
140 NEXT I
150 END
```

**3**

```
10 DIM A(3),B(3)
20 INPUT A(1),A(2),A(3)
30 INPUT B(1),B(2),B(3)
40 REM S=(A,B)
50 LET S=0
60 FŌR I=1 TŌ 3
70 LET S=S+A(I)*B(I)
80 NEXT I
90 REM A0=ABS(A),B0=ABS(B)
100 LET A0=0
110 LET B0=0
120 FŌR I=1 TŌ 3
130 LET A0=A0+A(I)^2
140 LET B0=B0+B(I)^2
150 NEXT I
160 LET A0=SQR(A0)
170 LET B0=SQR(B0)
180 REM CŌS(THETA)
190 LET X=S/(A0*B0)
200 REM FUNCIŌN ARCCŌS(X)
210 LET T=ATN(SQR(1-X^2)/X)
220 REM ŌUTPUT
230 PRINT T
240 END
```

**4**

プログラムを次のように改造するとよい。

```
10 REM EXAMPLE-2.6.3
20 DEF FNC(X)=1/(1+X*X)
30 INPUT "A,B,N=",A,B,N
40 LET W=0
50 LET W=0
60 FŌR I=1 TŌ N-1
70 LET X=A+I*H
80 LET W=W+FNC(X)
90 NEXT I
100 LET W=FNC(A)/2+W+FNC(B)/2
110 LET S=W*H
120 PRINT "S=";S;4*S
130 GŌTŌ 30
```

結果は次のようになる。

```
RUN
A,B,N=0,1,10
S= .784982  3.13993
A,B,N=0,1,20
S= .785294  3.14118
A,B,N=0,1,50
S= .785382  3.14153
A,B,N=0,1,100
S= .785394  3.14158
A,B,N=0,1,200
S= .785397  3.14159
```

**5**

行 10〜40 を次のように変更すればよい。

```
10 INPUT N¥
20 IF N¥="FŌRMULA-2" THEN 90
30 IF N¥="HERŌN" THEN 130
40 IF N¥<>"FŌRMULA-1" THEN 10
```

## 3.1 演習

**1**

```
10 PRINT "F1,I1=";
20 INPUT  F1,I1
30 PRINT "F2,I2=";
40 INPUT  F2,I2
50 LET L1=12*F1+I1
60 LET L2=12*F2+I2
70 LET L=L1+L2
80 LET F=INT(L/12)
90 LET R=L-12*F
100 PRINT F;"FEET";
```

```
110 PRINT R;"INCHES"
120 END
```

## 3.1 節の応用問題

1

```
10 INPUT N
11 IF N=1 THEN 70
12 IF N=2 THEN 50
20 FŌR M=2 TŌ SQR(N)
30 IF N/M=INT(N/M) THEN 70
40 NEXT M
50 PRINT "SŌSUU DEARU"
60 END
70 PRINT "SŌSUU DEWANAI"
80 END
```

2

```
10 DIM A(1000)
20 FŌR L=1 TŌ 1000
30 LET A(L)=0
40 NEXT L
50 FŌR K=2 TŌ 31
60 IF A(K)<>0 THEN 120
70 FŌR I=2 TŌ 1000/K
80 LET L=I*K
90 IF L>1000 THEN 120
100 LET A(L)=1
110 NEXT I
120 NEXT K
130 FŌR L=K TŌ 1000
140 IF A(L)=1 THEN 160
150 PRINT L
160 NEXT L
170 END
```

3

```
10 DIM P(25)
20 FŌR I=1 TŌ 25
30 READ P(I)
40 NEXT I
50 DATA 2,3,5,7,11,13,17,19
   ,23
60 DATA 29,31,37,41,43,47,5
   3,59
70 DATA 61,69,71,73,79,83,8
   9,97
80 INPUT N
90 IF N>100 THEN 80
100 PRINT N;"=";
110 FŌR I=1 TŌ 25
120 LET K=P(I)
130 IF K>N THEN 210
140 IF N/K<>INT(N/K) THEN 20
    0
150 PRINT K;
160 LET N=N/K
170 IF N=1 THEN 210
180 PRINT "*";
190 GŌTŌ 130
200 NEXT I
210 END
```

4

```
10 PRINT "A,B,C,D=";
20 INPUT A,B,C,D
30 LET F=B*C+A*D
40 LET E=A*C
50 LET S¥="+"
60 GŌSUB 300
70 LET F=B*C-A*D
80 LET S¥="-"
90 GŌSUB 300
100 LET F=B*D
110 LET S¥="*"
120 GŌSUB 300
130 LET E=A*D
140 LET F=B*C
150 LET S¥="/"
160 GŌSUB 300
170 END
300 PRINT
310 PRINT B;TAB(6);D;TAB(11)
    ;F
320 PRINT "__␣";S¥;"␣___␣=␣_
    ____"
330 PRINT A;TAB(6);C;TAB(11)
    ;E
340 PRINT
350 RETURN
360 END
```

5

```
10 PRINT "A,B,C,D="
20 INPUT A,B,C,D
30 LET R=A+C
40 LET I=B+D
50 LET S¥="+"
60 GŌSUB 300
70 LET R=A-C
80 LET I=B-D
90 LET S¥="-"
100 GŌSUB 300
110 LET R=A*C-B*D
120 LET I=A*D+B*C
130 LET S¥="*"
140 GŌSUB 300
```

```
150 LET R=(A*C+B*D)/(C*C+D*D)
160 LET I=(B*C-A*D)/(C*C+D*D)
170 LET S¥="/"
180 GŌSUB 300
190 END
300 PRINT "(";A;"+";B;"I)";
310 PRINT S¥;
320 PRINT "(";C;"+";D;"I)=";
330 PRINT "(";R;"+";I;"I)"
340 RETURN
350 END
```

**6**

```
10 DIM C(12)
20 DATA 0,31,59,90,120,151,181
30 DATA 212,243,273,304,334
40 FŌR I=1 TŌ 12
50 READ C(I)
60 NEXT I
70 INPUT "Y1,M1,D1=",Y,M,D
80 GŌSUB 300
90 LET A1=N
100 INPUT "Y2,M2,D2=",Y,M,D
110 GŌSUB 300
120 LET A2=N
130 LET A=A2-A1
140 PRINT A
150 END
300 LET Z=Y-1600
310 LET N2=365*Z+INT(Z/4)-INT(Z/100)+INT(Z/400)
320 LET N1=C(M)+D
330 LET N=N1+N2
340 IF Y/4<>INT(Y/4) THEN 390
350 IF Y/400=INT(Y/400) THEN 370
360 IF Y/100=INT(Y/100) THEN 390
370 IF M<=2 THEN 390
380 LET N=N+1
390 RETUN
400 END
```

**7**

文字列型配列を使用できる機種ならば次のようにすればよい(基本 BASIC には文字列型配列が含まれていないので，その制約を忠実に守るとすれば ON 文で分岐してプリントさせるほかない)。

```
10 DIN C(12),W¥(7)
20 DATA 0,31,59,90,120,151,181
30 DATA 212,243,273,304,334
40 DATA "SAT","SUN","MŌN","TUE"
50 DATA "WED","THU","FRI"
60 FŌR I=1 TŌ 12
70 READ C(I)
80 NEXT I
90 FŌR I=0 TŌ 6
100 READ W¥(I)
110 NEXT I
120 INPUT "Y,M,D=",Y,M,D
130 GŌSUB 300
140 LET R=N-7*INT(N/7)
150 PRINT W¥(R)
160 END
300
 ～   前問解答に同じ.
390
```

## 3.2 演習(p. 85)

**1**

```
10 DIM Y(60),M(12),D(31)
20 FŌR I=1 TŌ 60
30   LET Y(I)=0
40 NEXT I
50 FŌR I=1 TŌ 12
60   LET M(I)=0
70 NEXT I
80 FŌR I=1 TŌ 31
90   LET D(I)=0
100 NEXT I
110 FŌR I=1 TŌ 200
120   INPUT Y1,M1,D1
130   LET Y(Y1)=Y(Y1)+1
140   LET M(M1)=M(M1)+1
150   LET D(D1)=D(D1)+1
160 NEXT I
170 FŌR I=1 TŌ 60
180   PRINT I,Y(I)
190 NEXT I
200 FŌR I=1 TŌ 12
210   PRINT I,M(I)
220 NEXT I
230 FŌR I=1 TŌ 31
240   PRINT I,D(I)
250 NEXT I
260 END
```

**2**

```
10 LET K=0
20 INPUT T1
```

```
30 FŌR I=2 TŌ 50
40    INPUT T2
50    IF T1>=T2  THEN 70
60    LET K=K+1
70    LET T1=T2
80 NEXT I
90 PRINT K;"NICHI"
100 END
```

**3**

行100以後を次のように変更する。

```
100 DATA "HŌKKAIDŌU"
110 DATA "AŌMŌRI"
        ……以下同様……
550 DATA "KAGŌSIMA"
560 DATA "ŌKINAWA"
570 FŌR I=1 TŌ 47
580    READ A¥
590    PRINT A¥,K(I)
600 NEXT I
610 END
```

## 3.2　演習(p. 87)

**1**

(略)

**2**

```
10 DIM K(31,5)
20 FŌR I=1 TŌ 31
30    FŌR J=1 TŌ 5
40    LET K(I,J)=0
50    NEXT J
60 NEXT I
70 INPUT D,C,A
80 IF D=0 THEN  130
90 IF D>31 THEN  210
100 IF C>5 THEN  210
110 LET K(C,D)=K(C,D)+A
120 GŌ TŌ 70
130 PRINT "HIZUKE_BETU KATU
    HIMŌKU_BETU GŌUKEI"
140 FŌR I=1 TŌ 31
150    FŌR J=1 TŌ 5
160       PRINT K(I,J),
170    NEXT J
180    PRINT
190 NEXT I
200 STŌP
210 PRINT "IRENAŌSITE KUDAS
    AI"
220 GŌTŌ 70
230 END
```

## 3.2　演習(p. 89)

**1**

例題のプログラムに下記を挿入（一部変更）する。

```
221    LET G(I)=S(I)/N
222    LET W=0
223    FŌR J=1 TŌ N
226       W=W+(X(I,J)-G(I))^2
225    NEXT J
226    LET B(I)=W/N
230    PRINT "␣I␣";G(I),B(I)
291    LET H(I)=T(I)/M
292    LET W=0
293    FŌR I=1 TŌ M
294       W=W+(X(I,J)-H(I))^2
295    NEXT I
296    LET C(I)=W/M
321 FŌR J=1 TŌ N
322    PRINT C(I),
323 NEXT J
15 DIM G(5),H(5),B(5),C(5)
```

## 3.2　演習(p. 91)

(学生番号の記録はしないで，点数のみ)

**1**

```
10 INPUT N,X
20 LET S=X
30 FŌR I=2 TŌ N
40 INPUT "TUGI NŌ TEN=";X
50 IF S<=X THEN  70
60 LET S=X
70 NEXT I
80 PRINT "SAISHŌTI=";S,
90 END
```

**2**

```
10 INPUT N,X
20 LET S=X
30 LET G=X
40 FŌR I=2 TŌ N-1
50    INPUT "TUGI NŌ TEN=";X
60    IF G>=X THEN  90
70    LET G=X
80    GŌTŌ  110
90    IF S<=X THEN  110
100    LET S=X
110 NEXT I
120 PRINT "SAISHŌTI=";S,"S
    AIDAITI=";G
130 END
```

**3**

```
10  LET N=60
20  INPUT X
30  LET G=ABS(X)
40  FŌR I=2 TŌ N
50    INPUT X
60    LET A=ABS(X)
70    IF G>=A THEN 90
80    LET G=A
90  NEXT I
100 PRINT "MAX(ABS(X))=";G
110 END
```

**4**

```
10  LET N=60
20  INPUT S,X
30  IF S=2 THEN 60
40  LET G1=X
50  GŌTŌ 70
60  LET G2=X
70  FŌR I=1 TŌ N-1
80    INPUT S,X
85    IF S=2 THEN 120
90    IF G1>=X THEN 140
100   LET G1=X
110   GŌTŌ 140
120   IF G2>=X THEN 140
130   LET G2=X
140 NEXT I
150 PRINT "SAIKŌTEN","ŌTŌKŌ"
    ;G1,"ŌNNA";G2
160 END
```

**5**

```
10  DIM A1(3,5),B1(3,5),G1(3
    ),H1(3)
20  INPUT "NINZU=",M
30  FŌR G=1 TŌ 3
40  FŌR K=1 TŌ 5
50  LET A1(G,K)=1000000
60  NEXT K
70  NEXT G
80  PRINT "GAKUNEN,KUMI,BANG
    Ō,TIME"
90  FŌR I=1 TŌ M
100 INPUT G,K,B,T
110 IF A1(G,K)<=T THEN 140
120 LET A1(G,K)=T
130 LET B1(G,K)=B
140 NEXT I
150 FŌR G=1 TŌ 3
160 LET G1(G)=A1(G,1)
170 LET H1(G)=B1(G,1)
180 FŌR K=2 TŌ 5
190 IF G1(G)<=A1(G,K) THEN 2
    20
200 LET G1(G)=A1(G,K)
210 LET H1(G)=B1(G,K)
220 NEXT K
230 NEXT G
240 LET Y=G1(3)
250 LET Z=H1(3)
260 FŌR I=2 TŌ 1 STEP -1
270 IF Y<=G1(I) THEN 300
280 LET Y=G1(I)
290 LET Z=H1(I)
300 NEXT I
310 PRINT "ZENKŌU-1␣WA";Z;"K
    UN,";
320 PRINT "TIME";Y;"BYŌU"
330 FŌR G=1 TŌ 3
340 PRINT G;"GAKUNEN-1␣WA";H
    1(G);
350 PRINT "KUN,TIME";G1(G);"
    BYŌU"
355 NEXT G
360 PRINT "CLAS␣BETU␣NŌ␣1BAN
    "
370 FŌR G=1 TŌ 3
380 FŌR K=1 TŌ 3
390 PRINT G;"NEN";K;"KUMI␣";
400 PRINT B1(G,K);"KUN␣";A1(
    G,K);
410 PRINT "BYŌU"
420 NEXT K
430 NEXT G
440 END
```

**6**

データ $a_1, a_2, \cdots, a_n$ の最大値と 2 番目の値を求めるプログラムの例を示す。Tは最大値，Sは 2 番目の値である。

```
10  INPUT N,A1,A2
20  IF A1<A2 THEN 60
30  LET T=A1
40  LET S=A2
50  GŌTŌ 80
60  LET T=A2
70  LET S=A1
80  FŌR I=3 TŌ N
90  INPUT A
100 IF A<=S THEN 180
110 IF K<=T THEN 160
120 REM S<=T<A
130 LET S=T
140 LET T=A
150 GŌTŌ 180
```

```
160 REM S<A<=T
170 LET S=A
180 NEXT I
190 PRINT "TŌP";T
200 PRINT "SECŌND";S
210 END
```

## 3.2 節の応用問題

**1**

3.1節の応用問題の解答のプログラムの行130以後を次のように改造する。

```
130 FŌR J=1 TŌ 10
140 LET M1=100*(J-1)+1
150 LET M2=100*J
160 LET C=0
170 FŌR M=M1 TŌ M2
180 IF A(M)<>0 THEN 200
190 LET C=C+1
200 NEXT M
210 PRINT C;TAB(10);
220 FŌR I=1 TŌ C
230 PRINT "*";
240 NEXT I
245 PRINT
250 NEXT J
260 END
```

**2**

生徒番号0によって終了判定を行なうことにする。

```
10 DIM K(40)
20 FŌR I=1 TŌ 40
30 LET K(I)=0
50 NEXT I
60 REM SEITŌ BANGŌU Ō YŌMU
70 INPUT I
80 IF I=0 THEN 120
90 IF I>40 THEN 70
100 LET K(I)=K(I)+1
110 GŌTŌ 70
120 FŌR M=0 TŌ 366
130 PRINT "KESSEKI";M;"NITI"
140 FŌR I=1 TŌ 40
150 IF K(I)<>M THEN 170
160 PRINT I;
170 NEXT I
180 NEXT M
190 END
```

**3**

```
10 DIM K(9),M(9)
20 DATA 10000,5000,1000
30 DATA 500,100,50,10,5,1
40 FŌR L=1 TŌ 9
50 READ K(L)
60 LET M(L)=0
70 NEXT L
80 INPUT "NINZU=",N
90 FŌR I=1 TŌ N
100 PRINT "A(";I;")=";
110 INPUT A
120 FŌR L=1 TŌ 9
130 LET Q=INT(A/K(L))
140 LET A=A-Q*K(L)
150 LET M(L)=M(L)+Q
160 NEXT L
170 NEXT I
180 FŌR L=1 TŌ 9
190 PRINT K(L);"EN";TAB(10);
    M(L);"MAI"
200 NEXT L
210 END
```

**4**

```
10 LET S=0
20 LET N=0
30 REM LŌŌP
40 INPUT A
50 IF A=99999 THEN 100
60 LET S=S+A
70 LET N=N+1
80 GŌTŌ 40
90 REM MEAN
100 LET M=S/N
110 PRINT "HEIKIN␣";M
120 END
```

**5**

```
10 DIM X(1000),Y(1000)
20 LET W1=0
30 LET W2=0
40 INPUT N
50 FŌR I=1 TŌ N
60 INPUT X(I),Y(I)
70 LET W1=W1+X(I)
80 LET W2=W2+Y(I)
90 NEXT I
100 LET M1=W1/N
110 LET M2=W2/N
120 LET S1=0
130 LET S2=0
140 LET S3=0
150 FŌR I=1 TŌ N
160 LET S1=(X(I)-M1)^2+S1
170 LET S2=(Y(I)-M2)^2+S2
180 LET S3=(X(I)-M1)*(Y(I)-M
    2)+S3
```

```
190 NEXT I
200 LET S1=SQR(S1/N)
210 LET S2=SQR(S2/N)
220 LET R=S3/(N*S1*S2)
230 PRINT R
240 END
```

## 3.3.2 演習

1 (略)
2 (略)

## 3.3.5 応用問題

1

```
10 DIM M(20)
20 FŌR I=0 TŌ 20
30 LET M(I)=0
40 NEXT I
50 INPUT N
60 FŌR J=1 TŌ N
70 INPUT X
80 LET L=INT(X/5)
90 LET M(L)=M(L)+1
100 NEXT J
110 LET M(19)=M(19)+M(20)
120 FŌR I=0 TŌ 19
130 PRINT I*5;"-"; I*5+4;TAB
    (10);
140 IF M(I)=0 THEN 180
150 FŌR K=1 TŌ M(I)
160 PRINT "*";
170 NEXT K
180 PRINT
190 NEXT I
200 END
```

2

データは
年齢　男の数　女の数　（最後は年齢 999 とする）
の形式で入れ，ヒストグラムは 1 字で 1000 人を表すことにする。

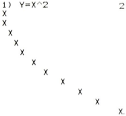
```
10 DIM A(9),B(9)
20 FŌR I=0 TŌ 9
30 LET A(I)=0
40 LET B(I)=0
50 NEXT I
60 REM LŌŌP
70 INPUT K,M,W
80 IF K=999 THEN 160
90 LET J=K/10
100 IF J<=9 THEN 120
110 LET J=9
120 LET A(J)=A(J)+M
130 LET B(J)=B(J)+W
140 GŌTŌ 70
150 REM ŌUTPUT
160 PRINT
170 FŌR J=9 TŌ 0 STEP -1
180 PRINT 10*J;TAB(5);
190 LET N1=INT(A(J)/1000+0.5
    )
200 IF N1<=20 THEN 220
210 LET N1=20
220 FŌR L=0 TŌ 20-N1
230 PRINT "␣";
235 NEXT L
240 IF N1=20 THEN 280
250 FŌR L=1 TŌ N1
260 PRINT "*";
270 NEXT L
280 PRINT "I";
290 LET N2=INT(B(J)/1000+0.5
    )
300 FŌR L=1 TŌ N2
310 PRINT "*";
320 NEXT L
330 PRINT
340 NEXT J
350 PRINT
360 END
```

3

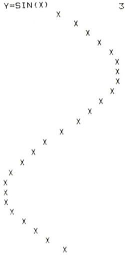
次ページの図を見よ。
上記 6）のために改造したプログラム

```
5 OPEN "O",#2,"LPT0:"
7 INPUT "TITLE",T$
8 PRINT #2,T$
10 DEF FNC(X)=1/(X+0.001)
20 PRINT "A,B,C,H,K="
30 INPUT  A,B,C,H,K
40 LET N=INT((B-A)/H+0.1)
50 FOR I=0 TO N
60 LET X=A+I*H
70 LET Y=FNC(X)
80 LET L=INT(C*Y+0.5)+K
81 IF L<=0 THEN 87
83 IF L>40 THEN 87
85 GOTO 90
87 PRINT #2
89 GOTO 130
90 FOR J=1 TO L
100 PRINT #2," ";
110 NEXT J
120 PRINT #2,"X"
130 NEXT I
135 CLOSE #2
140 END
```

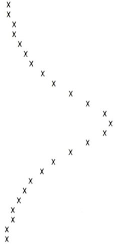
1)  Y=X^2



2)  Y=SIN(X)



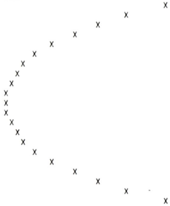
3)  Y=1/(1+X^2)



4)  Y=SQR(1-X^2)



5)  Y=COSH(X)



6)  Y=1/X





**4**

行20，行30，行50および行130を次のように変更し，$r(\theta)$を計算する関数FNR(T)を付加すればよい。

```
20 PRINT "N,T0,T1="
30 INPUT N,T0,T1
50 LET H=(T1-T0)/(N-1)
51 LET T=T0
52 LET R=FNR(T)
53 LET X(1)=R*CŌS(T)
54 LET Y(1)=R*SIN(T)
130 LET T=T0+H*(M-1)
131 LET R=FNR(T)
132 LET X(M)=R*CŌS(T)
133 LET Y(M)=R*SIN(T)
```

**5**

次のようなFNR(T)を用いればよい。

i)　FNR(T)=T

ii)　FNR(T)=SIN(5*T)

iii)　FNR(T)=CŌS(T)+1

結果は105ページの三つの図のようになる（N=100とした場合）。

## 3.4.3　応用問題

**1**

```
10 DIM A(3,3),B(3,3),C(3,3)
20 MAT READ A,B
30 MAT C=A+B
40 MAT PRINT C
50 DATA 6,1,8,7,5,3,2,9,4
60 DATA 2,5,6,9,4,7,8,1,3
70 END
```

（上記のデータは

$$A=\begin{pmatrix}6&1&8\\7&5&3\\2&9&4\end{pmatrix}\qquad B=\begin{pmatrix}2&5&6\\9&4&7\\8&1&3\end{pmatrix}$$

の場合である）。

**2**

```
20 MAT READ A(2,2)
30 MAT READ B(2,2)
40 MAT C=A*B
50 MAT PRINT C
60 MAT D=B*A
70 MAT PRINT D
80 DATA 3,2,5,1
90 DATA 2,9,6,5
100 END
```

**3**

```
10 DIM A(2,3),B(3,4)
20 MAT READ A,B
30 MAT C=A*B
40 MAT PRINT C
50 MAT E=TRN(A)
60 MAT F=TRN(B)
70 MAT G=F*E
80 MAT PRINT G
90 DATA 8,1,5,2,9,3
100 DATA 6,3,3,5,3,2,5,1,1,0
    ,9,3
110 END
```

**4**

```
10 DIM A(2,3),B(2,3)
20 MAT READ A,B
30 MAT C=TRN(A)
40 MAT D=C*B
50 MAT PRINT D
60 MAT E=TRN(B)
70 MAT F=A*E
80 MAT PRINT F
90 DATA 1,2,3,4,5,6
100 DATA 1,2,3,4,5,6
110 END
```

**5**

```
10 DIM A(3,3),X(3)
20 MAT READ A,X
30 MAT Y=A*X
40 MAT PRINT Y
50 DATA 9,8,7,6,5,4,3,2,1
60 DATA 3,6,7
70 END
```

**6**

```
10 DIM A(2,2),X(2),Y(2)
20 INPUT T
30 MAT READ X
40 LET A(1,1)=CŌS(T)
50 LET A(1,2)=SIN(T)
60 LET A(2,1)=-SIN(T)
70 LET A(2,2)=CŌS(T)
80 MAT Y=A*X
90 MAT PRINT Y
100 DATA 1,2
120 END
```

7

```
10  DIM A(20,20),B(
    20,20)
20  DIM X(20),Y(20)
    ,Z(20)
30  INPUT N
40  MAT READ A(N,N)
50  MAT READ B(N,N)
60  MAT READ X(N)
70  MAT Y=B*X
80  MAT Z=A*Y
90  MAT PRINT Z
100 DATA 2
110 1,2,3,4
120 5,6,7,8
130 9,10
140 END
```

(10, 20) 配列の寸法は十分大きくとっておく。

(40, 50) 行列の実際の次数は，こちらで決まる。DIM文の寸法には定数しか書けないがMAT READ文の寸法としては変数を書くことができる。ただしその値はDIM文で宣言した寸法を越えてはならない。

(120, 130) テスト用のデータ

8

```
10  DIM A(10,20)
20  INPUT M,N
30  MAT READ(M,N)
40  GŌSUB 800
50  END
60  DATA 2,3
70  DATA -1,0,1,2,
    -3,0
800 REM PATTERN PRINT
810 PRINT
820 FŌR I=1 TŌ M
830 FŌR J=1 TŌ N
840 LET C=A(I,J)
850 IF C<0 THEN 910
860 IF C=0 THEN 890
870 PRINT "+";
880 GŌTŌ 920
890 PRINT "0";
900 GŌTŌ 920
910 PRINT "-";
920 NEXT J
930 PRINT
940 NEXT I
950 RETURN
```

(10〜70) テスト用のメイン・プログラムとデータ

## 3.5.2 演習

```
10  INPUT N
20  LET Ō3=0
30  LET Ō=0
40  FŌR K=1 TŌ N
50  IF RND>0.5 THEN
    100
60  REM ŌTŌKŌNŌKŌ
70  PRINT "M";
80  LET Ō=0
90  GŌTŌ 150
100 REM ŌNNANŌKŌ
110 PRINT "W";
120 LET Ō=Ō+1
130 IF Ō<>3 THEN 150
140 Ō3=Ō3+1
150 NEXT K
160 PRINT "ZENBUDE";N;"NIN"
170 PRINT "ŌNNANŌKŌ␣GA␣3KAI␣
    TUZUKETE"
180 PRINT "UMARETA␣KAISUU␣WA
    ␣";Ō3
190 END
```

(10) 実験回数をNとする。

(20) 女の子が3人続けて生まれた回数をŌ3とする。

(30) 女の子が連続して生まれた人数のカウンター

(50) 乱数が0.5より大ならば女とする。

## 3.5.3 関数 RND の応用問題

1

```
10  LET I=INT(10*RND)
20  LET J=INT(10*RND)
30  PRINT I;"+";J;"=";
40  INPUT K
50  IF I+J=K THEN 80
60  PRINT "X"
70  GŌTŌ 90
80  PRINT "Ō"
90  GŌTŌ 10
100 END
```

2

```
10  DIM P(6),Q(6)
20  FŌR I=1 TŌ 6
30  READ P(I)
40  NEXT I
50  DATA 15,10,30,10,20,15
60  LET Q(1)=P(1)/100
70  FŌR I=2 TŌ 6
80  LET Q(I)=Q(I-1)+P(I)/100
90  NEXT I
100 LET R=RND
110 FŌR I=1 TŌ 6
120 IF R<=Q(I) THEN 140
130 NEXT I
140 PRINT I
150 STŌP
160 GŌTŌ 100
170 END
```

## 3.5.7 演習

1

(略)

### 4.1.3 応 用 問 題

**1**

文字列配列の使用が可能なら，次のようにするとよい。

```
10 DIM L¥(40)
20 FŌR X=0 TŌ 3 STEP 0.1
30 FŌR I=0 TŌ 40
40 LET L¥(I)="␣"
50 NEXT I
60 FŌR I=10 TŌ 30 STEP 10
70 LET L¥(I)="I"
80 NEXT I
90 LET K=20+10*SIN(X)
100 LET L¥(K)="+"
110 LET A=X
120 LET F=0
130 FŌR J=1 TŌ 5
140 LET F=F+A
150 LET K=20+10*F
160 IF K>40 THEN 190
170 IF K<0 THEN 190
180 L¥(K)="*"
190 LET A=-A*X*X/(2*J*(2*J+1))
200 NEXT J
210 FŌR I=0 TŌ 40
220 PRINT L¥(I);
230 NEXT I
240 PRINT
250 NEXT X
260 END
```

**2**

次のようなプログラムで数値的に出力し，手でプロットするのが安全。

```
10 FŌR X=0.3 TŌ 3.0 STEP 0.1
20 LET Y=(X-1)/(X+1)
30 LET F=0.86304*Y+0.3645*Y^3
40 LET E=F-LŌG(X)/LŌG(10)
50 PRINT X,F,E
60 NEXT X
70 END
```

**3**

```
10 LET C1=0.13451446
20 LET C2=2.6252949
30 LET C3=4.0333293
40 LET A3=1/3
50 LET A5=2/15
60 LET A7=7/315
80 FŌR X=0 TŌ 1 STEP 0.1
90 LET T=X*3.1416/4
100 LET Y1=(C1+C2/(C3-X*X))*X
110 LET Y2=SIN(T)/CŌS(T)
120 LET Z=T*T
130 LET Y3=(((A7*Z+A5)*Z+A3)*Z+1)*T
140 LET Y0=TAN(T)
150 PRINT X,Y0,Y1,Y2,Y3
160 NEXT X
170 END
```

計算時間の測定には，TIME（機種によってはTIM）という組み込み関数を使用するとよい。たとえば

```
10 LET C1=0.13451446
20 LET C2=2.6252949
30 LET C3=4.0333293
40 PRINT TIME
50 FŌR I=1 TŌ 1000
60 LET Y=(C1+C2/(C3-X*X))*X
70 NEXT I
80 PRINT TIME
90 END
```

**4**

```
900 REM ___ERRŌR FUNCTIŌN___
910 LET Y=(.078108*X+0.000972)*X
920 LET Y=(Y+0.230389)*X+0.278393
930 LET Y=1-1/(1+Y*X)^4
940 RETURN
```

**5**

```
800 REM ___BESSEL FUNCTIŌN ___
810 LET Z=(X/3)^2
820 LET Y=0.0002100*Z-0.0039444
830 LET Y=(Y*Z+0.0444479)*Z-0.3163866
840 LET Y=(Y*Z+1.2656208)*Z-2.2499997
850 LET Y=1+Y*Z
860 RETURN
```

**6**

```
600 REM __GAMMA(X)__
610 LET N=INT(X-1)
620 LET Z=X-N-1
630 LET Y=(-0.1010678*Z+0.4245549)*Z
640 LET Y=(Y-.6998588)*Z+.9512363)*Z
```

```
650 LET Y=(Y-.5748646)*Z+1
660 IF N=0 THEN 710
665 LET Z=Z+1
670 FŌR I=1 TŌ N
680 LET Y=Z*Y
690 LET Z=Z+1
700 NEXT I
710 RETURN
```

**7**

```
740 REM ___BETA(P,Q)___
745 LET X=P
750 GŌSUB 610
755 LET Y1=Y
760 LET X=Q
765 GŌSUB 610
770 LET Y2=Y
775 LET X=P+Q
780 GŌSUB 610
785 LET Y=Y1*Y2/Y
790 RETURN
```

## 4.2.4 応用問題

**1**

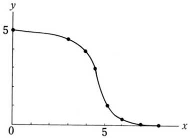
高度0~10 kmの標準大気密度$\rho$ (kg/m³)

```
10 DIM Y(10)
20 FŌR I=0 TŌ 10
30 READ Y(I)
40 NEXT I
50 DATA 1.2250,1.1117
60 DATA 1.0006,0.9093
70 DATA 0.8194,0.7364
80 DATA 0.6601,0.5900
90 DATA 0.5258,0.4671,0.4135
100 INPUT "X=",X
110 LET I=INT(X)
120 LET D=X-I
130 LET R=Y(I)+(Y(I+1)-Y(I))*D
140 PRINT "R=";R
150 END または GŌTŌ 100
```

**2**

125ページのプログラムに下記を追加する。

```
10 DIM X(10),Y(10)
20 FŌR I=1 TŌ 8
30 READ X(I),Y(I)
40 NEXT I
50 DATA 0,5
60 DATA 3,4.7
70 DATA 4,3.9
80 DATA 4.5,3
90 DATA 5.2,1
100 DATA 6,0.3
110 DATA 7,0.1
120 DATA 8,0
130 LET N=8
140 INPUT Z
```




本文のプログラムを挿入

```
300 LET S=(Z-X(L))/(X(L+1)-X(L))
310 LET F=Y(L)+(Y(L+1)-Y(L))*S
320 PRINT F
330 END または GŌTŌ 140
```

**3**

入力の部分は前問解答の行10~140と同じでよい。また，$x$含む区間をさがす部分は125ページのプログラムを使用できる。次数を3次（4点使用）に固定すれば，補間のプログラムは次のようになる。

```
300 IF L<2 340
310 IF L>5 360
320 LET K=L-1
330 GŌTŌ 370
340 LET K=1
350 GŌTŌ 370
360 LET K=N-3
400 LET P=0
410 FŌR I=K TŌ K+3
420 LET A=1
430 FŌR J=K TŌ K+3
440 IF I=J THEN 460
450 LET A=A*(Z-X(J))/(X(I)-X(J))
460 NEXT J
470 LET P=P+Y(I)*A
480 NEXT I
490 PRINT P
500 END または GŌTŌ 140
```

**4**

4次(5点)の Lagrange 補間を用いるとすれば次のようになる。

```
5 DIM X(4),Y(4)
10 FŌR I=0 TŌ 4
15 READ X(I),Y(I)
20 NEXT I
25 DATA 0,0
30 DATA 30,0.5
35 DATA 45,0.707107
40 DATA 60,0.866025
45 DATA 90,1
50 FŌR Z=0 TŌ 90
55 LET M=4
```

本文 127 ページのプログラム（行 100～180）を挿入

```
200 PRINT Z,P,SIN(Z),P-SIN(Z
)
210 NEXT Z
220 END
```

**5**

ヒントに従ってプログラムを作ると次のようになる。

```
10 DIM X(3),Y(3)
20 FŌR I=0 TŌ 3
30 READ X(I),Y(I)
40 NEXT I
50 DATA 1,1
60 DATA 2,1
70 DATA 3,2
80 DATA 4,6
90 FŌR Z=2 TŌ 3
```

本文127ページのプログラム（行100～180）を挿入
ただし，Mの所は3とする。

```
200 LET F=P/(Z-1)
210 PRINT Z-1,F
220 NEXT Z
230 END
```

### 4.3.3 応 用 問 題

**1**

先に4.3.1節で示した Simpson の公式のプログラムを用いると，結果は次のようになる。

i)
$N=10$ で計算すると，$S=0.2525$
$N=100$ で計算すると，$S=0.250025$
$N=1000$ で計算すると，$S=0.25$
(単精度で計算した場合)

ii)
$N=100$ で計算すると，$S=1.7183\cdots$

iii)
$N=100$ で計算すると，$S=0.785394\cdots$

**2**

プログラムと結果

```
10 LET N=10
20 LET S=0
30 FŌR A=1 TŌ 2.1 STEP 0.1
40 LET B=A+0.1
50 PRINT A,S
60 LET H=(B-A)/N
70 LET W=0
80 FŌR I=1 TŌ N-1
90 LET X=A+I*H
100 LET W=W+1/X
110 NEXT I
120 LET W=(1/A)/2+W+(1/B)/2
130 LET S=W*H+S
140 NEXT A
150 END
```

```
 1              0
 1.1            9.53117E-02
 1.2            .182324
 1.3            .262368
 1.4            .336476
 1.5            .40547
 1.6            .470009
 1.7            .530634
 1.8            .587793
 1.9            .64186
 2              .693154
```

**3**

```
10 DEF FNC(X)=EXP(-X^2)
20 INPUT "A,B,N=",A,B,N
30 LET H=(B-A)/N
40 LET  W=0
50 FOR I=1 TO N-1
60 LET X=A+I*H
70 LET W=W+FNC(X)
80 NEXT I
90 LET W=FNC(A)/2+W+FNC(B)/2
100 LET S=W*H+S
110 PRINT S
120 END

Ready
RUN
A,B,N=0,5,10
 .886227

-
```

**4**

プログラムは前問と同じでよい。関数 FNC (X) および実行結果は次のとおり。

```
10 DEF FNC(X)=SQR(1-X^2)

Ready
RUN
A,B,N=0,1,100
 .785104
```

### 4.4.2 演習

1

上のプログラム例の行110以後を次のように変更すればよい。

(1)

```
110 LET F=-X*Y
120 RETURN
```

入力データ（$\varDelta x=0.1$ で $x=1$ まで計算する場合）

```
0,1,0.1,10
```

(2)

```
110 LET F=X*X/Y
120 RETURN
```

入力データ（$\varDelta x=0.05$ で $x=1$ まで計算する場合）

```
0,1,0.05,20
```

(3)

```
110 LET F=Y-SIN(X)
120 RETURN
```

入力データ（$\varDelta x=0.1$ で $x=5$ まで計算する場合）

```
0,0.5,0.1,5
```

2

$-1\leqq y\leqq 1$ の範囲のグラフを描くのであれば，上のプログラム例の行40を次のように修正すればよい。

```
40 PRINT X,Y,TAB(50+10*Y);"
   *"
```

なお，その前に，オーバースケールの防止のため，

```
39 IF ABS(Y)>1 THEN 50
```

を挿入するとよい。

### 4.4.4 演習

1

(略)

### 4.4.7 応用問題

1

```
10 DIM Y(2),YO(2),F(2)
20 DIM F1(2),F2(2),F3(2)
30 READ XO,N,H,K
40 FOR I=1 TO N
50 READ YO(I)
60 Y(I)=YO(I)
70 NEXT I
80 FOR L=0 TO K
90 X=XO+L*H
100 GOSUB 500
110 FOR I=1 TO N
120 F1(I)=F(I)
130 Y(I)=YO(I)+F(I)*H/2
140 NEXT I
150 X=X+H/2
160 GOSUB 500
170 FOR I=1 TO N
180 F2(I)=F(I)
190 Y(I)=YO(I)+F(I)*H/2
200 NEXT I
210 GOSUB 500
220 FOR I=1 TO N
230 F3(I)=F(I)
240 Y(I)=YO(I)+F(I)*H
250 NEXT I
260 X=X+H/2
270 GOSUB 500
280 FOR I=1 TO N
290 Y(I)=YO(I)+(F1(I)+2*F2(I)
    +2*F3(I)+F(I))*H/6
300 YO(I)=Y(I)
305 IF L<>INT(L/10+0.01)*10
    THEN 320
310 PRINT Y(I),
320 NEXT I
325 IF L<>INT(L/10+0.01)*10
    THEN 340
330 PRINT X;TAB(60+Y(1)*5);"
    *"
340 NEXT L
350 STOP
500 REM ---びぶんけいすう の けいさん---
510 F(1)=Y(2)
520 F(2)=-SIN(Y(1))
530 RETURN
535 REM ---  データー  ---
540 DATA 0,2,0.1,300
550 DATA 3.0
560 DATA 0
570 END
```

2 (略)

### 4.5節と4.6節の応用問題

1

i)

$f(x)=2x-\tan x$

$f'(x)=2-\sec^2 x$

であるから，

$$x_{k+1}=x_k-\frac{2x-\tan x_k}{2-1/\cos^2 x_k}$$

を反復すればよい。

```
10 LET X=1
20 FOR K=1 TO 10
30 LET F=2*X-TAN(X)
40 LET D=2-1/COS(X)^2
50 PRINT K;X,F
60 LET X=X-F/D
70 NEXT K
80 END
```

ii）

出発値は鍵盤から入力することにする（根が2箇あることに注意）。

```
10 INPUT X
20 FŌR K=1 TŌ 10
30 LET F=2*X-(EXP(X)+EXP(-X
   ))/2
40 LET D=2  -(EXP(X)-EXP(-X
   ))/2
50 PRINT K;X,F
60 LET X=X-F/D
70 NEXT K
80 END
```

iii）

上と同様。ただし

```
30 LET F=X*X-SIN(X)
40 LET D=2*X-CŌS(X)
```

**2**

```
10 DEF FNS(X)=(EXP(X)-EXP(-
   X))/2
20 DEF FNC(X)=(EXP(X)+EXP(-
   X))/2
30 LET T=2
40 FŌR K=1 TŌ 10
50 LET F=1.1*T-FNS(T)
60 LET F=1.1*T-FNS(T)
60 LET D=1.1-FNC(T)
70 LET T=T-F/D
80 NEXT K
90 LET A=100/T
100 LET W=A*(FNC(100/A)-1)
110 PRINT "T=";T
120 PRINT "W=";W
130 END
```

**3**

厳密解は

i）

1, 2, 3

ii）

2, −1, $i$, $-i$

iii）

1（2重根）, −1（3重根）

## 4.7節と4.8節の応用問題

**1**

厳密解は

i）$x=-1/6$　$y=-1/2$　1

ii）$x=1/6$　$y=1/2$　$z=1$

**2**

厳密解は

i）$\begin{pmatrix} 119/12 & -7/2 & -5/3 \\ -28/12 & 1 & 1/3 \\ -19/12 & 1/2 & 1/3 \end{pmatrix}$

ii）$\begin{pmatrix} 5 & 7 & 6 & 5 \\ 7 & 10 & 8 & 7 \\ 6 & 8 & 10 & 9 \\ 5 & 7 & 9 & 10 \end{pmatrix}$

iii）$\begin{pmatrix} 2 & -1 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & -1 & 1 \end{pmatrix}$

**3**

157〜158ページのプログラムを次のように改造すればよい。

```
105 LET D=0              挿入
122 LET D=-D
150 LET D=D*A(K,K)
230 LET D=D*A(N,N)       変更
240 PRINT "DET=";D
```

下記の行を削除。

80, 90, 200, 250〜310, 123〜124

**4**

157ページのプログラムの行10〜100を次のように変更すればよい。

```
5 DIM A(3,3),X(3),B(3)
10 LET N=4
15 FŌR I=1 TŌ N
20 FŌR J=1 TŌ N
25 LET A(I,J)=0
30 NEXT J
35 LET B(I)=0
40 NEXT J
45 M=5
50 FŌR K=1 TŌ M
55 READ I,J,R
60 LET A(I,J)=-1/R
65 LET A(J,I)=-1/R
70 LET A(I,I)=A(I,I)+1/R
75 LET A(J,J)=A(J,J)+1/R
80 NEXT K
85 READ I,I0
```

```
90 LET B(I)=I0
92 FOR J=1 TO N
94 LET A(I,J)=0
96 NEXT J
98 LET A(I,I)=1
100 DATA 1,2, R1
101 DATA 1,3, R2
102 DATA 2,3, R3
103 DATA 2,4, R4
104 DATA 2,5, R5
105 DATA 1,   i0
```

(抵抗値，電流値を記入)

## 4.9 節の応用問題

**1**

厳密な固有値，固有ベクトルは次のようになる。

i） $\lambda_1=10$ $\lambda_2=5$ $\lambda$ $\lambda_3=2$ $\lambda_4=1$

$$\boldsymbol{v}_1=\begin{bmatrix}2\\2\\1\\1\end{bmatrix}\quad \boldsymbol{v}_2=\begin{bmatrix}-1\\-1\\2\\2\end{bmatrix}\quad \boldsymbol{v}_3=\begin{bmatrix}0\\0\\-1\\1\end{bmatrix}\quad \boldsymbol{v}_4=\begin{bmatrix}-1\\1\\0\\0\end{bmatrix}$$

ii） $\lambda_1=4$ $\lambda_2=40$ $\lambda_3=400$ $\lambda_4=4000$

$$\boldsymbol{v}_1=\begin{bmatrix}1\\1\\1\\1\end{bmatrix}\quad \boldsymbol{v}_2=\begin{bmatrix}1\\1\\-1\\-1\end{bmatrix}\quad \boldsymbol{v}_3=\begin{bmatrix}1\\-1\\-1\\1\end{bmatrix}\quad \boldsymbol{v}_4=\begin{bmatrix}1\\-1\\1\\-1\end{bmatrix}$$

（注意） 固有ベクトルの計算結果は，普通，上記と同じにならないが，成分の比が上記に一致すればよい。

**2**

165 ページのプログラムの行 50〜80 を次のように変更すればよい。

```
50 LET A(I,J)=0
55 NEXT J
60 LET A(I,I)=2
65 IF I=N THEN 80
70 LET A(I,I+1)=-1
75 LET A(I+1,I)=-1
80 NEXT I
```

**3**

165 ページのプログラムの行 10〜80 を次のように変更すればよい。

```
10 DIM A(3,3),V(3,3)
15 INPUT K1,K2
20 INPUT I1,I2,I3
25 LET A(1,1)=K1/I1
30 LET A(1,2)=-K1/SQR(I1*I2
)
35 LET A(1,3)=0
40 LET A(2,1)=A(1,2)
45 LET A(2,2)=(K1+K2)/I2
50 LET A(2,3)=-K2/SQR(I2*I3
)
55 LET A(3,1)=0
60 LET A(3,2)=A(2,3)
65 LET A(3,3)=K2/I3
70 LET N=3
```

また行 510 の次に挿入

```
515 PRINT "F=";SQR(A(J,J))/6
.2832
```

# 索　引

さ　行

英字索引

## 付録索引 ——FORTRAN——

**著者略歴**

**戸　川　隼　人**
と　がわ　はや　と

1935年　東京に生まれる
早稲田中学，早大高等学院を経て
1958年　早稲田大学第一理工学部数学科卒，
科学技術庁航空技術研究所研究員
初年度は東大に出向して真空管式
計算機ＴＡＣの開発に従事。以後，
数値計算法，計算機科学（図形処
理，数式処理，文献検索等），ロケ
ットの軌道計算，自動計測処理シ
ステム，構造解析等の研究を行な
う。1965年より計算第１研究室長
として計算センタ運営にあたる。
1971年　京都産業大学理学部計算機科学科
助教授（72年教授）。数値解析学
およびシステム工学を担当
1976年　日本大学理工学部数学科教授

**主要著書**
数値計算入門　FORTRANによる有限要素法入門
演習 FORTRANとその応用　有限要素法概論
数値解析とシミュレーション　共役勾配法

パソコン ライブラリ＝1
**基本BASICとその応用**

1982年 1月10日 ©　初 版 発 行
1990年 4月25日　初版第8刷発行

著　者　戸　川　隼　人　　発行者　森　平　勇　三
印刷者　林　　初　彦

発行所　**株式会社　サイエンス社**
〒101　東京都千代田区神田須田町2丁目4番地
安部徳ビル
〔営業〕☎（03）256-1091㈹　振替　東京7-2387
〔編集〕☎（03）256-1093㈹

印刷・製本　太洋社
《検印省略》



ISBN4-7819-0316-9
PRINTED IN JAPAN

ISBN4-7819-0316-9 C3341 P1906E

定価1,906円(本体1,850円)